SIMLOG : A LOGIC SIMULATOR

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

R. S. RAGHUNANDAN

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

JANUARY, 1987

CERTIFICATE

This is to certify that the thesis entitled 'SIMLOG : A LOGIC SIMULATOR' submitted by Mr. Raghunandan R.S., for the partial fulfilment of the requirements for the award of M.Tech. degree, has been carried out under my supervision. According to my knowledge, it has not been submitted elsewhere for an award of a degree.

R. Raghuram

5/1/87

(R. Raghuram)
Assistant Professor
Department of Electrical Engineering
Indian Institute of Technology
Kanpur.

ACKNOWLEDGEMENTS

It gives me great pleasure to express my sincere thanks and deep sense of gratitude to Dr. R. Raghuram for the splendid guidance and regular encouragement he has offered for completing my work.

I shall always remember the company of my friends R. Prasad and Jaya Raghavendra who always lent their helping hands whenever needed, at various stages of my work.

I also offer my thanks to Udaya, Madhusudhan and many other friends because of whom I enjoyed my stay at I.I.T., Kanpur.

Finally, I thank Mr. C.M. Abraham for his excellent service in typing the thesis.

Raghunandan R.S.

R.S.R

ABSTRACT

One of the important stages in any circuit design is logical verification of feasibility of the circuit designed. With this in view, the logic simulator 'SIMLOG' has been developed.

The simulator SIMLOG is written in PASCAL, includes a variety of features. It is a four valued simulator, for which circuits can be described using subcircuit facility, which eases circuit description. The elements that can be described to SIMLOG can have any arbitrary delay. The simulation of transmission gates is possible, enabling the simulation of NMOS circuits. It is actually possible to simulate unidirectional gates, bus elements and Ratio node logic for bidirectional gates.

SIMLOG makes use of linked type data structure of PASCAL for efficient simulation. The simulation module is based on 'Table Driven Activity Oriented Technique', which also contribute towards the saving in time.

As an example, a NMOS microprocessor bit slicer is designed and the design is verified using SIMLOG. The results, including simulation of this chip and simulation of same simple circuits having unit delay elements and arbitrary delay elements are also given.

The program consists of 6 Sections. They are Input description module, Timing scheme, Initial analysis, Scheduler, Evaluation and output modules.

LIST OF FIGURES

Note : Figures 5.2, 5.3, 5.5, 5.6, 5.8, 5.10 and 6.11 are included in Appendix B.

CHAPTER 1

INTRODUCTION

Logic Simulation is a process of representing a model of a digital circuit to a computer, and then evaluating signal values in the modelled circuit as a function of time for some applied input sequences.

The digital circuits can be modelled either mathematically or functionally. But in most of the cases, the complexity of circuit model, does not permit an exact mathematical definition. Hence functional methods in simulation is employed more often. Also in the functional model, there is a better correspondence between the model and the real system. This makes it a better method to generate test schedules or verify intuitive designs.

Simulation can be carried out at a number of levels.

(a) The highest and most general is the SYSTEMS LEVEL, which involves describing model and timing of the system in terms of high level languages like GPSS [10] and SIMSCRIPT [11]. This level is useful with large systems which can be partitioned into a number of subsystems and peripherals.

(b) The second level of simulation is at the REGISTER TRANSFER LEVEL. This models the flow of data at register level.

The modelling of microcode in computer and microprocessor type structures makes use of register, level Simulation as in AHPL [12].

(c) The level at which most of simulation is done in an IC design is at logic or gate level. At this level, actual gates and their interconnections are specified. The operations of gates are modelled and time is quantised in terms of gate propagation delays. LAMP [13] is an example of such a simulation system.

1.1 AIMS IN IMPLEMENTING A SIMULATOR

The major use of simulation of logic circuits is to verify and check designs both at gate level and the system level prior to actual manufacture. In a way this is the link between paper work and printed circuit board or IC layout.

Unlike the process of manufacturing PCBs, it is impossible to make minor changes to an LSI or VLSI circuit without repeating the expensive and time consuming operations of maskmaking and wafer fabrication. Also breadboard simulation cannot be carried out for LSI and VLSI circuits. Hence it is essential to simulate the operations of such circuits, before they are transformed into a layout and actual fabrication of chip is started.

Circuit simulation programs such as SPICE [14] have proven as vital computer aided design tools for the analysis

of the electrical performance of integrated circuits. SPICE can perform a variety of analyses including dc, ac and time domain transient analysis of circuits. But such a circuit simulator is cost-effective for analysing circuits of less complexity. For example, a 700 MOSFET circuit, analysed with an average 2-ns time step takes approximately 4 cpu hours on a VAX-11/780 VMS computer with floating point accelerator hardware [17].

Gate level logic simulators can verify circuit function in less time than a circuit simulator. Also digital circuit simulation usually does not require such critical analysis as existing in a circuit simulator, and simulation at gate level is often sufficient. But critical parts such as sense amplifier of a Dynamic RAM, it may be necessary to perform accurate electrical simulation. In such cases, SPICE may be used. The logic simulator helps to obtain information about logical correctness, timing and signal propagation characteristics including race and hazard conditions. The operations involved in a logic simulator are non-numeric.

1.1.1 Design Process

For designing a fully customised circuit, there are many stages which have to be considered in detail. The block diagram for such a design is shown in Figure 1.1 [5].

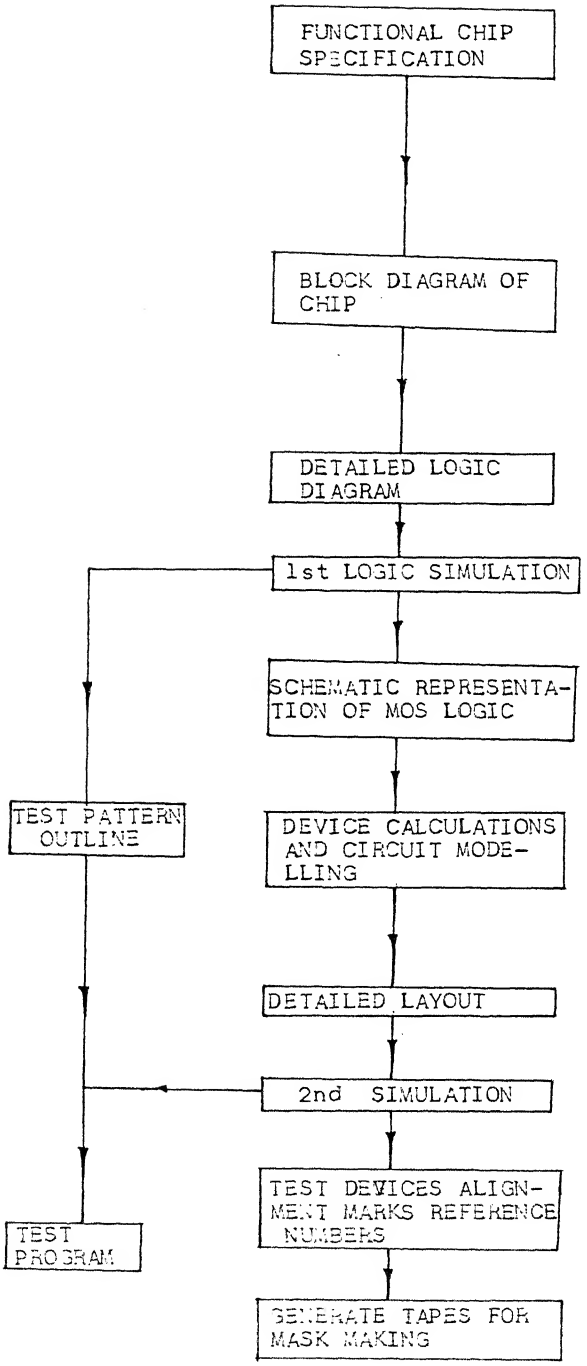


Figure 1.1 DESIGN PROCESS

The initial stage is to determine overall specifications and technology to be used. Though at the outset it looks easy, the definition of functional specification, of a proposed circuit always cause some difficulty.

The next important stage is to specify complete operation of the chip in block diagrams. This is the stage where one can get good idea about the architecture and size of the chip.

Then comes the logic diagram block where each block diagram is considered separately by a detailed logic diagram performing the required function. These logic circuits are important and naturally depend on the technology chosen.

At the next stage, CAD logic simulation program is essential, to verify the logic of different circuits. Also the data required to generate test programs can be obtained from this stage. This thesis explains the implementation of this stage to obtain logic simulator SIMLOG. Later, Chapter 5, gives the first 3 stages, where a NMOS microprocessor chip is designed and simulated.

Next a transistor level representation is obtained from the symbolic logic diagrams. One of the main characteristics of this stage is to obtain topological layout of individual circuit elements.

Then transistor sizes are calculated and circuit functions are modelled. A detailed layout will be done such that it will

map directly at a reduced scale onto the silicon circuit. A further simulation can be done to ascertain the circuit connections in the final laid out form.

The occurrence of an error in layout, either from logical or physical point of view, cannot be ruled out. It is understandable, because of the nature of complexity of circuits used. Recently CAD tools have been developed to check design rule violations or functional errors. These can be used to verify overall chip function. Some test transistors are included by which semiconductor processing characteristics are checked. Then the necessary alignment and identification marks are done before a file is generated in suitable format to interface with mask making equipment. 5 to 15 different mask layers are required for most common processes [5].

In parallel with the design of chip, the test program should be generated so that final chip will be testable. It will be necessary to modify the design to take account of testing requirements of circuit or limitations of test hardware available.

1.2 OVERVIEW OF A LOGIC SIMULATOR

The usefulness of logic simulation can be evaluated with respect to various attributes such as :

a) Accuracy : A close correspondence between predicted signal value vs time as given by simulator and that occurs in the actual circuit.

b) Efficiency: The simulation process must be cost effective.

c) Generality: Ability to handle broad class of circuits including :

(i) Combinational, synchronous and asynchronous circuits.

(ii) A wide selection of flip-flops, gates, RAMS and ROMS

(iii) Unknown signal values which may occur during initialisation or 'power up' of a circuit.

The essential parts of a logic simulation system, are shown in Figure 1.2.

(a) Simulation Description : This consists of input values at the start of simulation, input data or test sequence to be simulated, faults to be simulated and outputs to be monitored.

(b) Machine Description : This consists of the circuit to be modelled, described in terms of its components.

(c) Execution : This consists of

(i) Determining the logical values of gates and elements from the machine description.

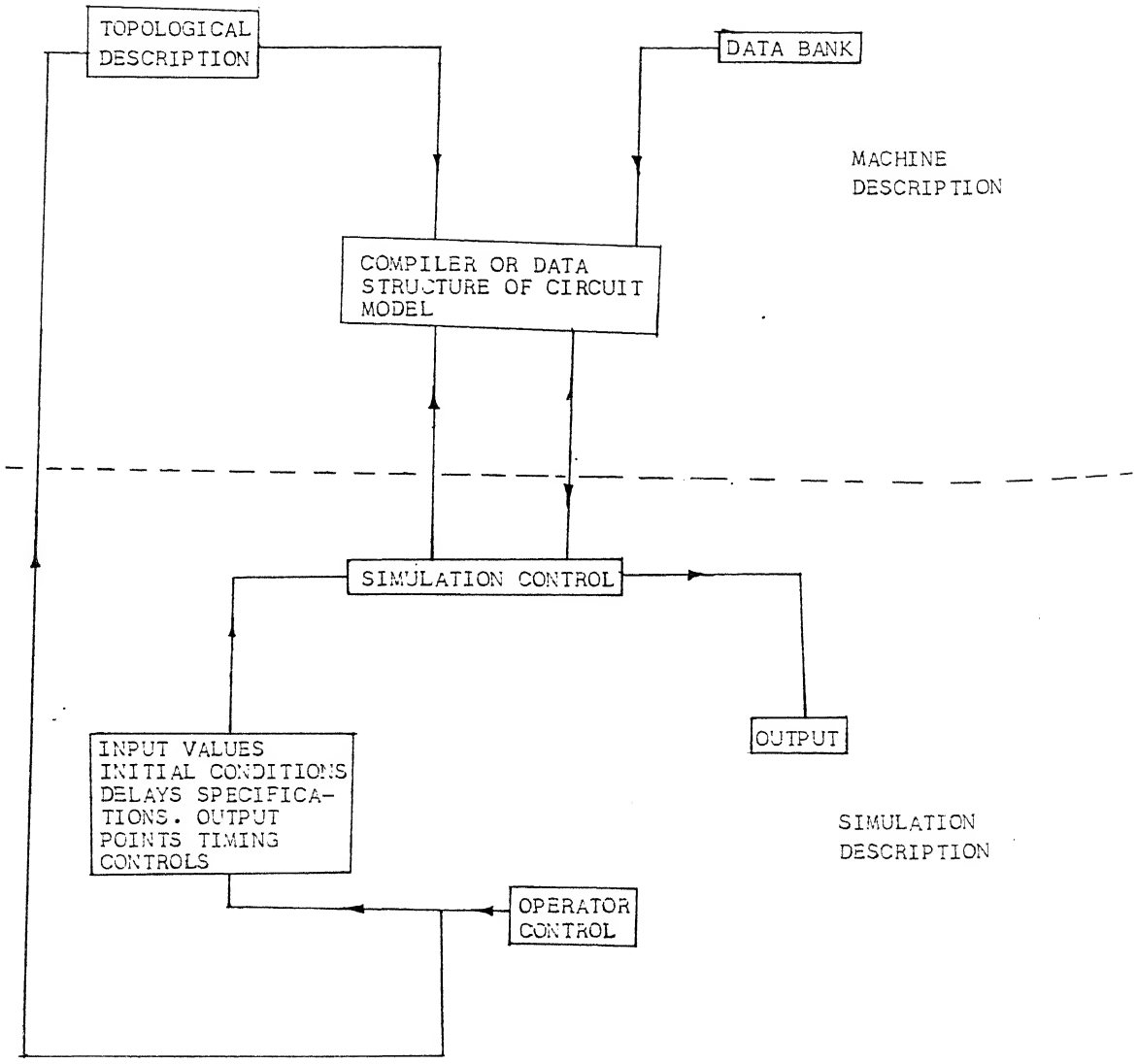


Figure 1.2 A LOGIC SIMULATION SYSTEM

(ii) Incorporating the changed values in the concerned fanout elements at correct time steps.

(iii) Printing the information both before and after simulation.

Some of these concepts regarding logic level simulator are described below. The primitives in such a simulator are elements.

The input information to a logic simulator usually consists of :

(a) Description of circuit to be simulated.

(b) Input data to be simulated.

(c) Initial value of memory states.

(d) Faults to be simulated, if any.

(e) Signals to be monitored.

The circuit description includes the topology of circuit and circuit element types, along with primary inputs and primary outputs. The specification of some other parameters like delay, fanin would also be included. The process of describing a circuit in terms of the primitives of the processing system is known as modelling. The main object is to model the circuit such that the simulator will give results corresponding to signal values in the actual circuit.

There are two classical ways of representing a circuit during simulation :

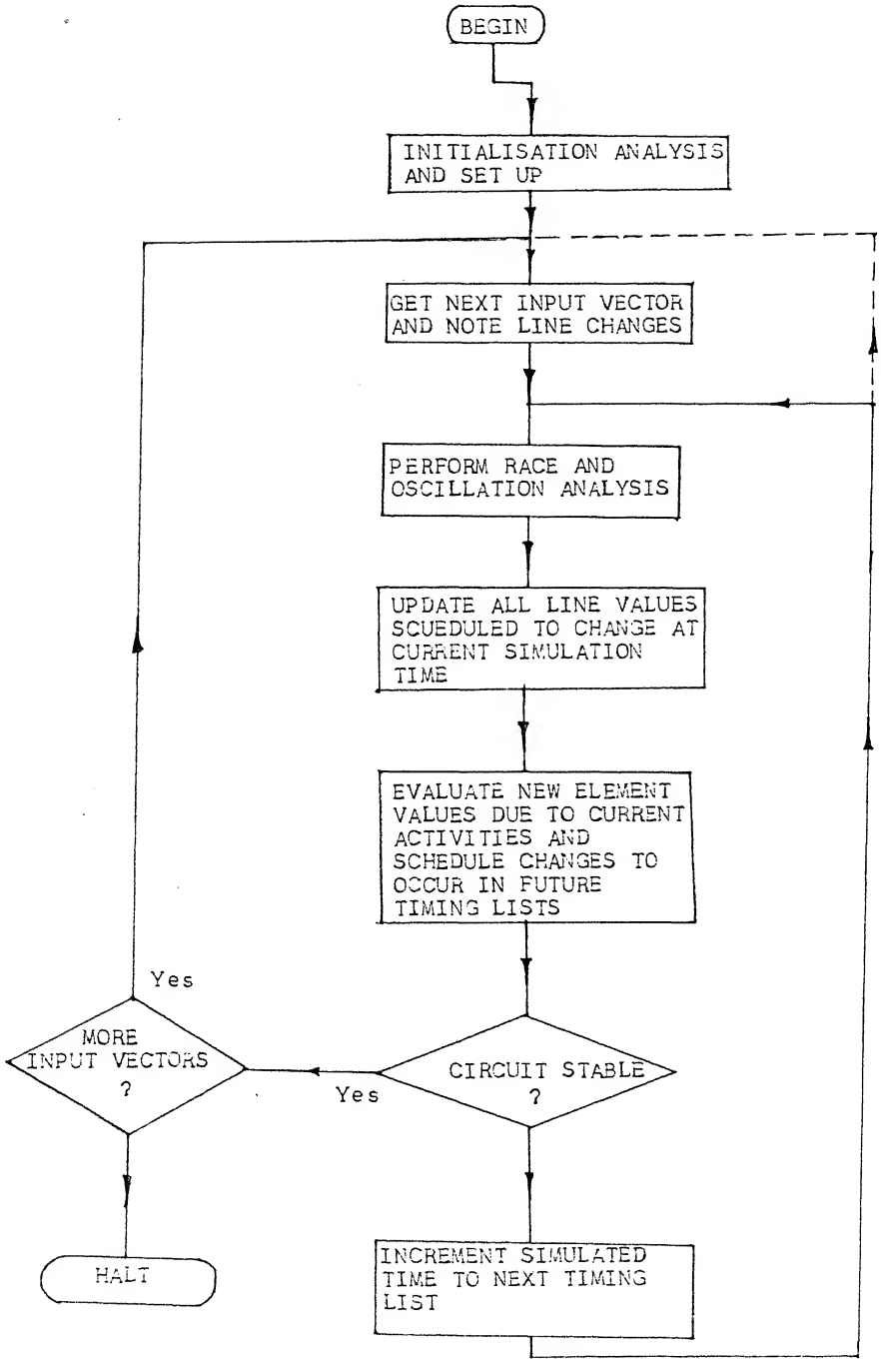


Figure 1.3 SIMULATE MODULE

(i) Compiler code structure, and

(ii) Table structure.

The function of these structure modules is to prepare internal data structure required for efficient simulation, based on input description.

The simulation model makes use of above data structure to get data and simulate signal changes. In a simulation system, time is quantized into units and simulation is done for each unit.

One of the simulate modules for a simulator is depicted in Figure 1.3 [1].

1.3 THESIS OUTLINE

The following chapters deal with the design, implementation and performance of logic simulator SIMLOG and also design of NMOS bit slicer chip.

Chapter 2 deals with defining different simulate modules used in SIMLOG. It also gives the data structure used in SIMLOG.

Chapter 3 gives the different features of SIMLOG.

Chapter 4 gives the details of program implementation in terms of flowcharts and explains the functions of different modules.

Chapter 5 gives the results obtained by simulating some typical circuits.

Chapter 6 gives the design and simulation of a NMOS bit slicer chip.

Chapter 7 draws conclusions by noting different advantages and limitations of SIMLOG, finally suggesting scopes for improving it.

Appendix A gives a complete manual for SIMLOG, to help and guide the user in creating input data file for simulator.

CHAPTER 2

SIMLOG - THE SIMULATOR

SIMLOG is a 4 valved simulator implemented in PASCAL on DEC 1090. The simulate module structure and data structure used in SIMLOG are described in this chapter.

The efficiency of any logic simulation system depends on the structure of its simulate module and sequence in which simulation proceeds. Logic simulation is a slow process when compared to working of an actual circuit, the reason being, while a simulator process elements sequentially, the signals can propagate simultaneously along diverse paths in an actual circuit. Hence reduction in simulation time is most desirable, especially for simulating large circuits.

As already mentioned there are two basic classes of simulators, compiler driven and table driven event directed. The earliest simulators were of the former type. But the latter type of simulation, will be more versatile in handling delays as well as reducing simulation time.

2.1 COMPILER DRIVEN SIMULATION

In this type of simulation, the structure module translates

the description of the circuit into machine executable code, such as an assembly language program.

Before the code is generated, the elements in a circuit are ordered. The first step in ordering is called levelizing. The second step is sorting.

(i) Levelising : This assigns a level number to each element and signal line. The logic level of any element is more than the highest logic level among its inputs. Also logic level values are assigned according to following rules :

(a) All primary input lines and feedback lines are placed at level O.

(b) For any element that has all its inputs already assigned logic levels, add 1 to the highest input logic level and assign this to the element.

(ii) Sorting : After levelling, every element is sorted in ascending order.

In sorting, all elements at the same level are grouped together. Sorting in ascending order is important since only then can it be ensured that all input to an element are evaluated before the evaluation of the element itself is taken up.

(iii) Generation of Code : Having ordered, the elements, the code generator outputs codes for each operations.

Taking an example of a NAND GATE, D with inputs A,B and C, the following code might be generated [1].

CODE	COMMENTS
LAC A	Load accumulator with A
ANA B	Form A AND B in the accumulator
ANA C	Form A AND B AND C in the accumulator
NTA	Negate the contents of accumulator
STR D	Store the above result in D.

Similarly, codes can be generated for AND, OR, NOR, latches, flip flops etc. These generated codes are typically stored in a sequential file with beginning level i labelled by L_i. The above code will be processed by simulate module.

In a circuit of N elements, the total time taken for simulation will be tN where t is the average simulation time for 1 element. This time t to simulate an element can be reduced, when output can be determined by just one of its inputs (dominating input). But one has to trade off this saving in time with length of the compiled code which will increase. The reason for that being usage of branch instruction codes.

One of the examples of compiler driven simulators is a compiler developed for AHPL at Department of Computer Science, IIT Kanpur. AHPL(Hardware Programming Language) has been

developed by Hill and Peterson [12]. It is a powerful hardware description language.

The simulation is done by first converting the AHPL description to PASCAL. Then the PASCAL compiler is used to run this machine generated code. User description of the hardware circuit can, therefore, be partly in PASCAL. This is convenient for input-output features of the circuit.

2.2 TABLE DRIVEN EVENT DIRECTED SIMULATION

It has been observed that during a single simulation computation scan only a small percentage of the signal lines change values. Usually the ratio of lines which change values to the total number of lines in the circuit, called 'activity' is between 2 to 10%. Thus simulating all elements in a circuit during every scan is thus wasteful. Selective trace or event (activity) directed simulation tries to reduce the simulation time by making use of above fact.

When there is a change in the signal line value, it is called as an event. When an event occurs, all elements to which this line fansout are placed on 'potentially active' list. In next simulation time, only such potentially active elements should be evaluated. Among these, some lines will change value and some others will not change. Fanouts of changed elements form the new 'potentially active' list. Such

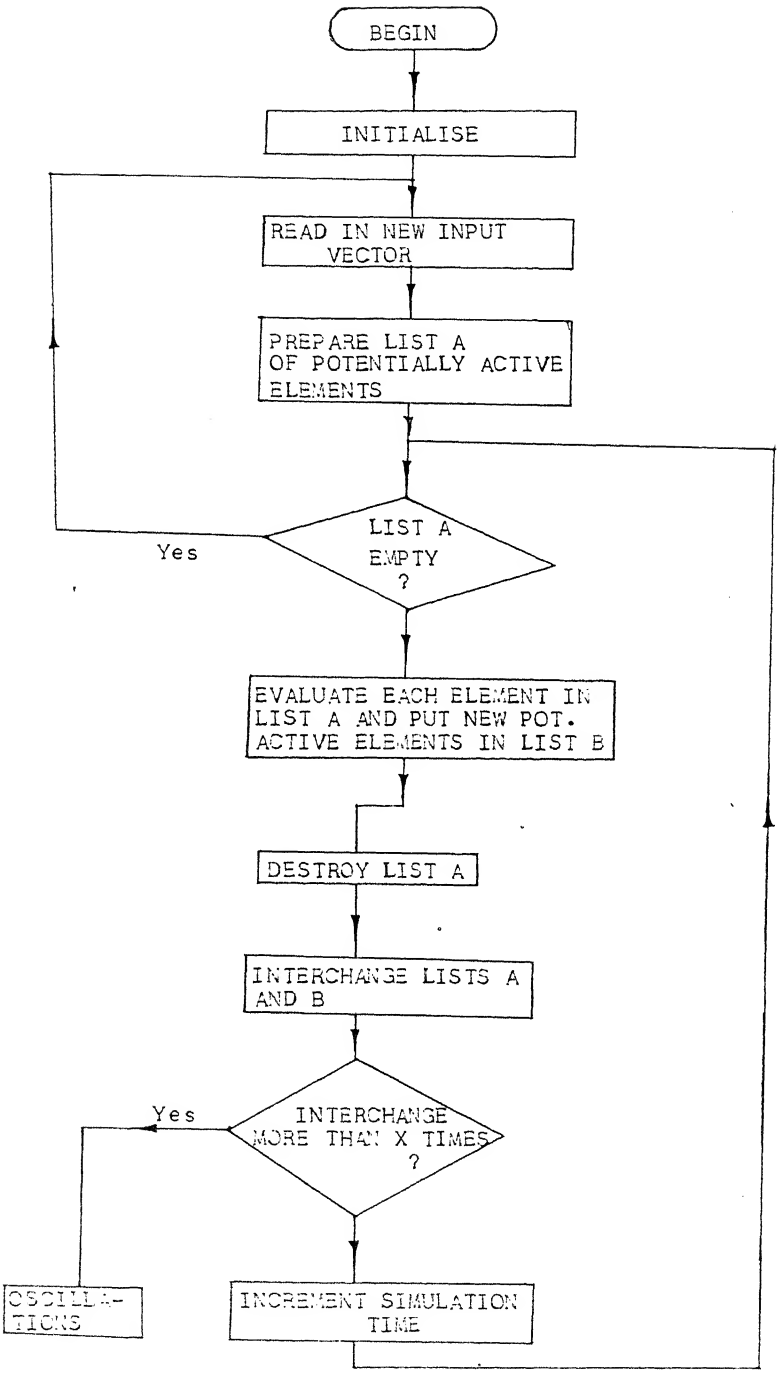


Figure 2.1 BASIC STRUCTURE OF A UNIT DELAY EVENT DIRECTED SIMULATOR

a simulator is capable of handling combinational, synchronous and asynchronous circuits with equal ease. Delays can also be processed in a simple manner.

The basic structure of a unit delay event directed simulator is shown in Figure 2.1 [15]. Such simulation requires special data structures which give different data information of each element.

2.3 CHOICE OF SIMULATOR TECHNIQUE IN SIMLOG

In the preceding sections, some simulating modules were analysed. The computer model of a logic circuit is formed from source language input described by the user. These input statements are transformed into machine code instructions in the case of a compiler driven simulator and into a data structure representing the circuit in table driven simulator case.

In the former case, a set of subroutines to perform logical function of each element in the circuit is generated. This is then executed in hierarchy defined by the logic level assigned to each element until output is reached. But this type of simulator has 2 major disadvantages.

(i) Each time a change occurs in the circuit, the entire process of code generation has to be repeated.

(ii) Simulation proceeds sequentially from input to output point, even through the paths that remain unactivated. This is wasteful in terms of computational time.

But table driven technique can reduce the computational time as explained in previous section. So this technique of table driven event directed simulation has been used in SIMLOG. The source language input statements are used to build a data structure which gives information of the elements in the circuit like function (logical), fanin, fanout, delay etc. During simulation, the control program in accordance with the command statements, manipulates information available in the table to obtain the logical value of each element at every instant.

2.4 DATA STRUCTURE USED IN SIMLOG

The data structure as proposed in Ref. [1] has been used in SIMLOG with modifications. The data structure is a simple linked list, with each element of a circuit represented as a record of linked list.

The element descriptor record is depicted in Figure 2.2a. The different fields of the record are :

a) INDEX : Each element and signal is represented by an integer value called Index.

b) NAME : Each element and signal has an alpha numeric name consisting of 4 letters.

c) VALUE : This is the logic value of the element. This can take 4 values

(i) 0 or 1 - known logic values

(ii) 2 - unknown logic value

(iii) 3 - high impedance state.

d) NFO : Number of fanouts of the element

e) NFI : Number of fanins of the element

f) FOL [i] : The index number of ith fanout of the element

g) FIL [i] : The index number of ith fanin of the element

h) KIND : Logical type of the element

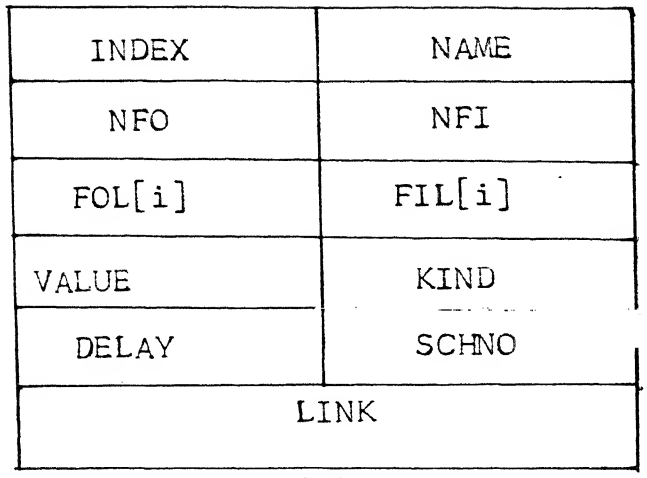
i) DELAY : This is the propagation delay of the element

j) SCHNO : Depicts whether the element is going to be evaluated or has already been evaluated.

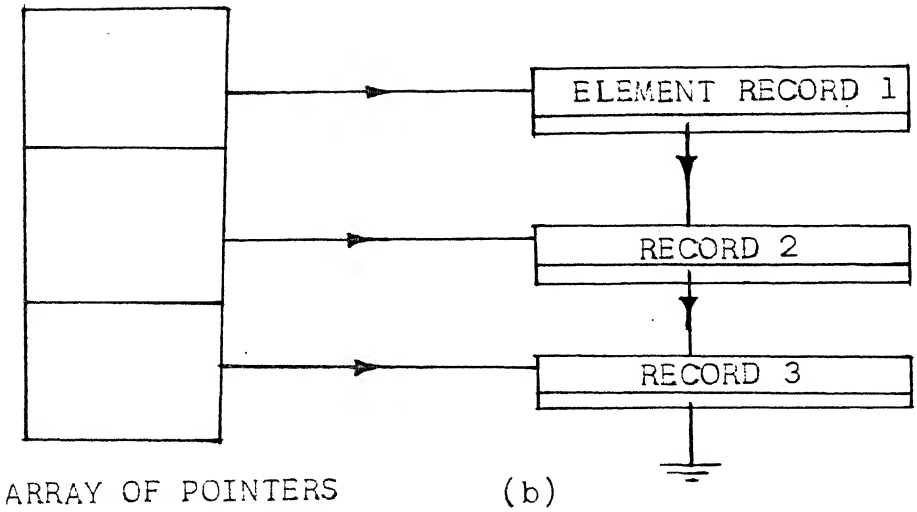
k) LINK : Links the present element descriptor record with next element descriptor record.

The Figure 2.2b depicts the data structure which has been used to access the required element descriptor record.

As shown, each unit of an array of pointers points to one element descriptor record. So by accessing the unit of array, we can access the element record and other fields of that record. By making use of this, the problem of searching through the linked list everytime, when an element has to be accessed, is avoided and thus saving in computational time.



(a)



(b)

Figure 2.2 DATA STRUCTURE USED IN SIMLOG

CHAPTER 3

FEATURES OF SIMLOG

As already mentioned, SIMLOG is a 4 valued simulator. The simulator can simulate various logic gates which include AND, OR, NOT, NAND, NOR and EXOR gates. The gates can have a maximum fanin of 10 and a maximum fanout of 20 gates/elements. Each gate/element will be represented by a unique Index number.

The features that simulator SIMLOG supports include :

a) 4 valued simulator
b) Subcircuit facility for describing circuits
c) Arbitrary delay for gates/elements
d) Provision for simulating transmission gates including bidirectional gate
e) Bus element provision is also provided.

This chapter describes in detail each of these features as implemented in SIMLOG.

3.1 SUBCIRCUIT FACILITY

The subcircuit facility greatly simplifies the process of describing large digital circuits in which usually, some blocks appear more than once. SIMLOG has only one level of subcircuit facility. Therefore, description of subcircuits inside a subcircu is not allowed.

The circuit description by the user is in a fixed format specified in the user's manual (Appendix A). In the main circuit description, an element is specified by its Index number, name, number of fanins, fanin's Index number, type of element and delay. All subcircuits which will be used should be described at the beginning of input file.

While describing subcircuits the first line should contain the name of the subcircuit, the number of external nodes and local Index numbers of external nodes, with some other information as described in user's manual. The description of other local elements follows the first line. This is analogous to that of the main circuit. Only, element numberings and hence Index numbers are local, and global linkages with main circuit are missing.

Each subcircuit description is stored in a separate linked list of element descriptor records, with one record each for every local number. Description of Input nodes, to the subcircuit, is optional. But every Input node should have one local Index number.

A global linked list is formed from the description of the main circuit that follows the description of all subcircuits. The records in this list have global Index numbers.

3.1.1 Subcircuit Expansion and Global Linking

Subcircuit calls will follow the main circuit description. When a subcircuit call is encountered, the subcircuit linked lists are searched, with the subcircuit name as key, to obtain the pointer to the corresponding subcircuit description. Then the linking of gates/elements of the subcircuit being called (other than input nodes), to the global linked list should be carried out.

The above processing involves :

(i) Reading the external global Index numbers provided by the user and assigning them to corresponding external local index numbers, to an array.

(ii) Completely filling the above array by providing Global index numbers to other elements/gates of the subcircuit and assigning the global Index number to corresponding local Index number.

(iii) Forming element descriptor records for every element/gate other than input nodes and then copying global index number from above array, corresponding to local Index number into the Index field. The remaining information is copied from the corresponding record in the subcircuit linked list. Here the fanin Index numbers for each gate will also be converted into corresponding global number.

(iv) Linking above records to global linked list and forming only one globally linked list.

3.2 CIRCUIT DELAY

Simulation deals with evaluating the logic value of each element as a function of time. Delays with in the circuits must be considered in order to correctly perform simulation. Delay is an important feature to the correct functioning of many circuits. For some elements such as delay lines, multivibrators, delay is the important characteristic of their operation. Therefore it is essential model the delay of circuit.

In general circuit delays can be modelled in more than one way.

a) Transport Delay ($\triangle_T$) : Every element and wire introduces delay to the propagating signals, when signal propagates through the element. This delay is represented by $\triangle_t$. If an element has transport delay $\triangle_t$, then if an input change occurs at time t, the effect of that change on output will occur only at time $t + \triangle t$.

b) Ambiguity Delay ($\triangle_M - \triangle_m$) : When an exact transport delay is not know, delay can be modelled as ambiguity delay. The delay through an element may in fact vary. In this case a pair of delay values are assigned to the element, $\triangle_m$ which will be minimum and $\triangle_M$ which will be maximum delay through that element. Thus these delays will define an ambiguity region of duration $(\triangle_M - \triangle_m)$.

c) Rise and Fall Delay ($\triangle_R, \triangle_f$) : The output response rise (0 to 1 change) and fall (1 to 0 change) time are different for some devices due to various electrical parameters such as capacitance. Such devices are modelled by assigning 2 different delays

$\triangle_R$ and $\triangle_f$ to the element at output. If $\triangle_R$ is not equal to $\triangle_f$, naturally, the pulse width propagating through element gets modified.

d) Inertial Delay ($\triangle_I$) : All devices used in a circuit will require energy in order to switch state. The minimum duration for which an input change must persist in order for the device to switch states is termed as inertial delay. It is denoted by $\triangle_I$.

Logic simulators can process delay in two ways namely delay propagation and min/max path analysis.

For delay propagation, signals are propagated through an element based upon their current inputs and their delay values ($\triangle_T$) as shown in Figure 3.1. For min/max path analysis, signals are first propagated using delay propagation technique, with delay parameters simplified, e.g., $\triangle_T = (\triangle_M + \triangle_m)/2$. Once this process is over critical circuit portions, such as 0 to 1 and 1 to 0 transitions are studied in detail [1].

In SIMLOG only transport delay $\triangle_t$ has been modelled and delay propagation technique is used.

3.2.1 Time Flow Mechanism and Event Scheduling

When an element i is simulated and it is determined. hat its value has changed a $V(j) \neq V'(j)$ then the line i must be scheduled to change to $V'(i)$ at time $t_o + \triangle$, if t_o is the current simulated time. $\triangle$ is the transport delay of the element i. In this case i is a scheduled event element and this event must be scheduled in the space for future events. This technique of scheduling and unscheduling events is called the time flow mechanism.

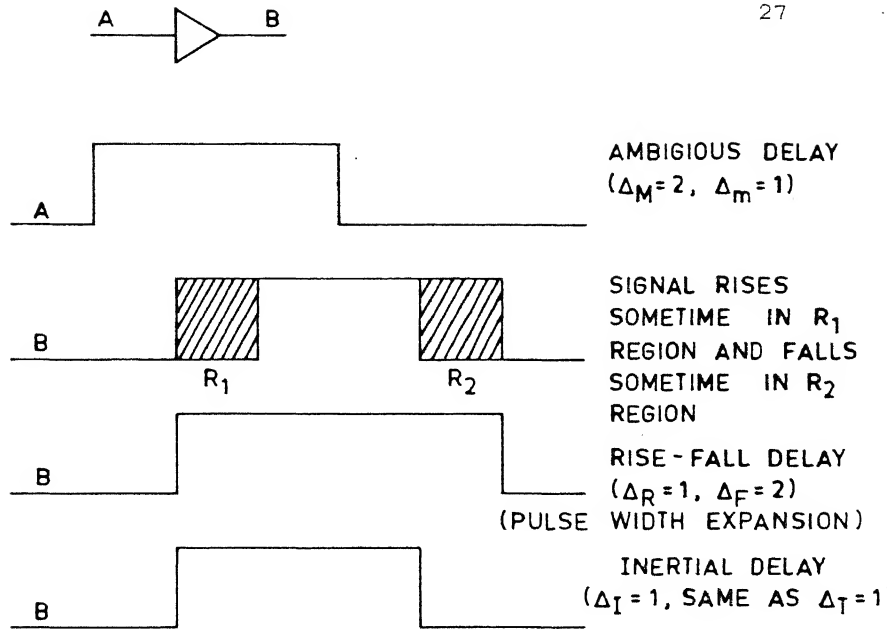


FIGURE 3.1. INPUT/OUTPUT TIMING DIAGRAMS DEPICTING VARIOUS DELAY MODELS.

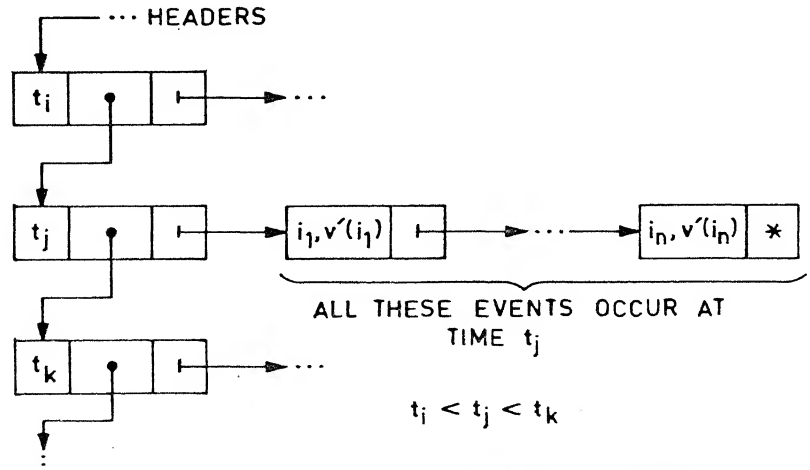


FIGURE 3.2. EVENTS SCHEDULER USING A LINKED LIST STRUCTURE.

One way to mechanise the scheduling of events is using a list structure as shown in Figure 3.2 [1]. Events occurring at same time t are stored in the same list. The problem with this mechanism is that for every occurring event, the list must be searched to get time, t from header, and then searched horizontally to append the event to that list. This consumes a lot of simulation time.

Another method is to use timing wheel scheduler where headers are stored sequentially and event time can be easily found out by $t+\triangle$. But overflow will occur whenever delay is greater than number of headersin timing wheel. Also another disadvantage is that some event lists may be empty.

A third method is to use a stack T, an array into which events to be scheduled will be placed. Then during time t, all events will be searched through stack T to schedule current events. Here also searching takes time. But here since it is an array, it becomes simpler to enter the events to be scheduled.

SIMLOG uses a stack T to schedule the events. The flow-chart used is shown in Figure 3.3.

If delay of an element inside stack is greater than 1, then during current time, delay is decreased by 1. When delay becomes 1, at time t, that event will be scheduled during time t

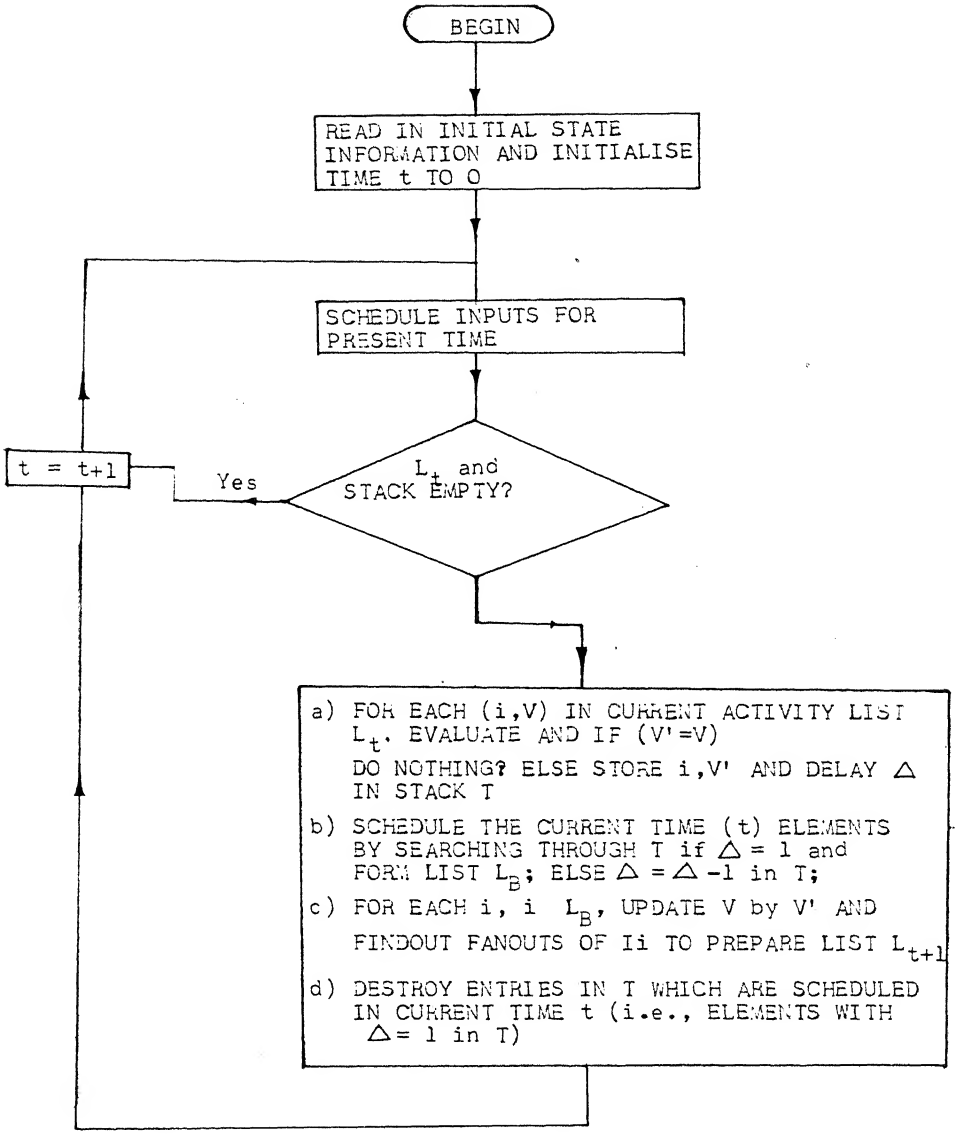


Figure 3.3 EVENT DIRECTED SIMULATOR ALGORITHM FOR ARBITRARY DELAY

and the old value of that element will be replaced by new value. This ensures that fanouts of elements that are scheduled will be placed in new potentially active list.

Also if a fanout element has delay 0, it should be scheduled in current time itself and it will not enter the stack T.

Thus if an element j entered stack at time t and element has a delay $\triangle$, then that event will be scheduled at $L_t + \triangle$ where L is potentially active list.

3.3 TRANSMISSION GATE

The 4 values that an element can take are 0 (low), 1(HIGH), 2 (UNKNOWN) and 3 (HIGH IMPEDANCE). Hence transmission gates can also be easily simulated.

The operation of a transmission gate is logically equivalent to a voltage controlled switch. When the controlling signal is ON, the switch is closed, connecting the input to output; when control signal is off, the switch is open, resulting in high impedance state. The truth table for a transmission gate is shown in Table 3.1 [6].

The output of one or more tri-state devices are typically tied to a 'bus', i.e., a wired connection of each input. Thus an element called BUS is also introduced and this can be simulated. This follows the BUSLOGIC of Table 3.2.

CONTROL (GATE)

INPUT	0	1	2	3
0	3	0	2	2
1	3	1	2	2
2	3	2	2	2
3	3	3	2	2

Table 3.1 TRANSMISSION GATE

INPUT A

INPUT B	0	1	2	3
0	0	0	0	0
1	0	1	1	1
2	0	1	2	2
3	0	1	2	*

Table 3.2 BUS LOGIC

* RETAINS PREVIOUS STATE

INPUT A

INPUT B	0	1	2	3
0	0	1	2	0
1	0	1	2	1
2	0	1	2	2
3	0	1	2	3

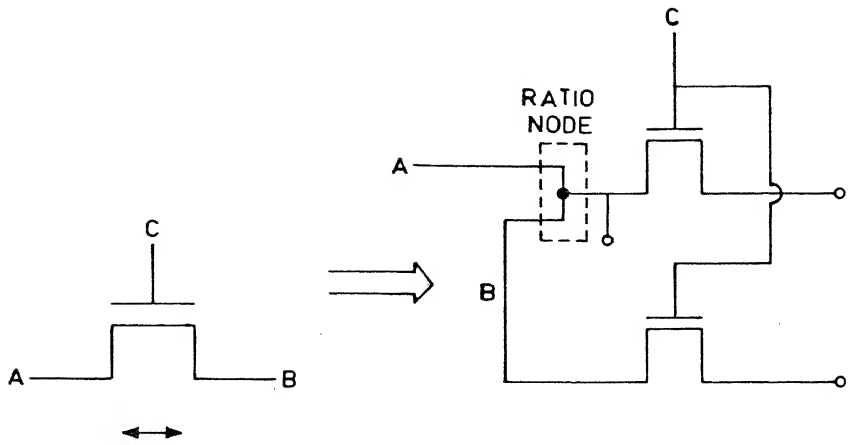
Table 3.3 RATIO LOGIC NODE (SIGNAL A STRONGER THAN SIGNAL B)

A problem which has received much attention recently is the modelling of bi-directional transmission gates. If the nodes on both sides have same value, there is no problem. Also if one of the nodes is in high impedance, it is clearly known which side is driving and which will be driven. If both sides of this gate, are driven to different known logic values, it is unclear as to the resultant values on either side. To avoid this problem, one side of a passtransistor must be known, apriori, to be of higher drive capability. Then other side will be driven to that value as the value at the driving node.

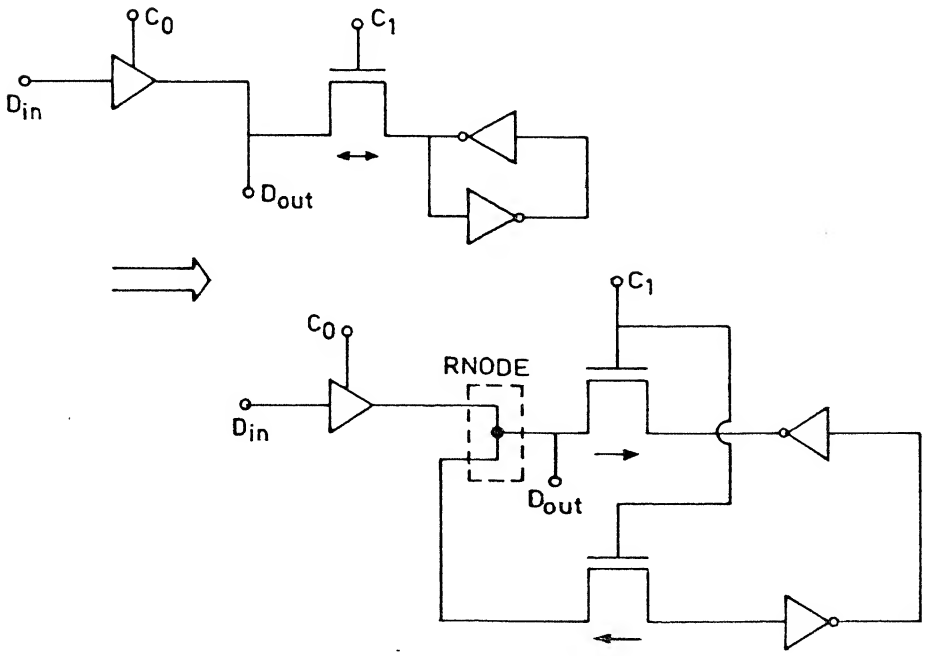
But one more problem is that existing selective trace table is inadequate to model the bidirectional gate. But a new model for bidirectional gate, using two back to back connected unidirectional transmission gate along with Ratio Logic node is used [6]. This is depicted in Figure 3.4a. The truth table for Ratio Logic node is given in Table 3.3.

Figure 3.4b shows an example making use of a bidirectional gate and its equivalent circuit.

The only other point to be noted is that for a BUS element, when inputs have 0 and 1, a bus conflict occurs. But in SIMLOG, the element is made to take value 0 in such cases, i.e., 0 is stronger than 1.



(a) BIDIRECTIONAL GATE.



(b) RAMCELL EXAMPLE USING BIDIRECTIONAL GATE MODEL.

FIGURE 3.4

CHAPTER 4

IMPLEMENTATION DETAILS

The program is divided into 6 modules based on their functions :

1) Circuit description module : This consists of procedures SUBCIRCUIT and READTABLE

2) Timing Scheme : This consists of TWOPHASECLK and CHANGECLK procedures.

3) Initialisation Module : This consists of the INITIALANAL procedure.

4) Scheduler Module : This consists of the SCHEDULER procedure

5) Evaluate Module : This consists of procedures which evaluate different gates and also procedure EVALUATE which is the heart of SIMLOG.

6) Output Module : This module include procedures PRINTTABLE, PRINTRES and PRINTLINE.

The main program calls some of these procedures in an orderly manner to achieve the logic simulation of a given circuit. Other procedures will in turn be called by these procedures.

4.1 CIRCUIT DESCRIPTION MODULE

4.1.1 Procedure Subcircuit

This procedure reads the description of elements of subcircuits and forms linked lists. Each subcircuit description forms one linked list. The first record of each list is called subcircuit record. The subcircuit record of first linked list (of first subcircuit described) is connected to subcircuit record of second linked list, second is connected to third and so on till last linked list is reached.

The flow chart is given in Figure 4.1. First the information regarding the subcircuit which includes the name of subcircuit, total number of elements in subcircuit, total number of external nodes and total number of input nodes are read. These are read into a record called SUBCIRCUIT. The first element descriptor will be linked to this record. The remaining element descriptors will be linked to first element to form a list. When the second subcircuit description is encountered the above procedure is repeated. But SUBCIRCUIT record of first list will also be connected to SUBCIRCUIT record of second list, as already explained.

Thus SUBCIRCUIT records of each subcircuit description also form a linked list. The element descriptor record is depicted in Figure 2.2.

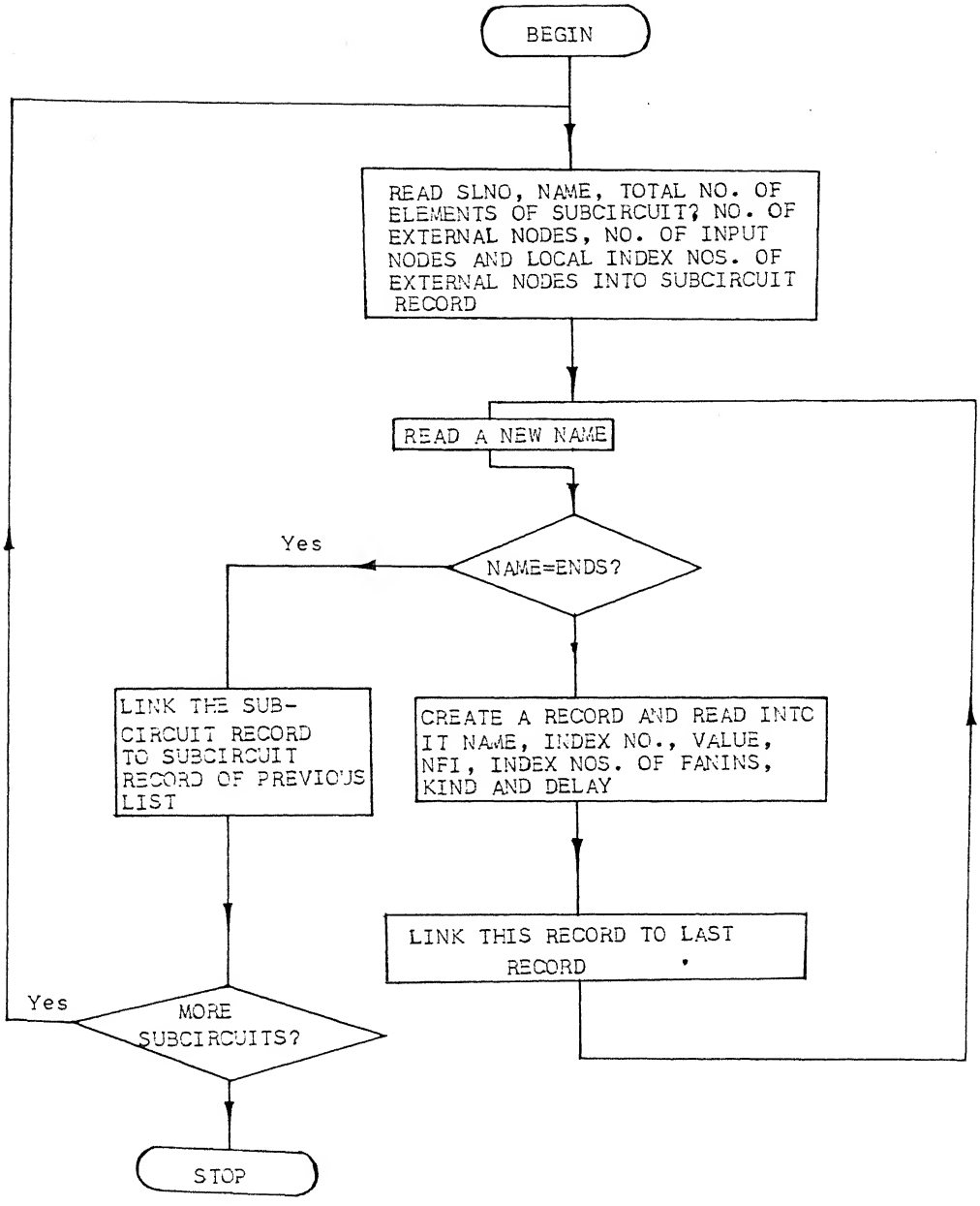


Figure 4.1 SUBCIRCUIT FLOWCHART

4.1.2 Procedure Read table

The flow chart of READTABLE is as shown in Figure 4.2. As already described here, all the global numbered elements except those which are output of some subcircuit will be described initially. Then subcircuit calls are made. The linking of called subcircuit elements to the global list will be done in this procedure.

Once the global list is formed, it is arranged in ascending order of index numbers. Then each element descriptor record is pointed to by a pointer of an array P.

4.2 TIMING SCHEME

4.2.1 Procedure Two Phase Clock

This procedure just forms two records, to put in information about 2 phased clocks PHI1 and PHI2. These two are nonoverlapping clocks derived from the main clock of circuit. When the main clock has a value 0, these two clocks also will be 0. If main clock is 1, one of PHI1 and PHI2 will be 1 and other being 0. But when clock goes high (1) again, the first one will become 0 and latter will have a value 1. This is depicted in Figure 4.3(a).

PHI1 record will have an Index number TOTAL+1 and PHI2 will have TOTAL+2 where TOTAL is total number of globally numbered elements.

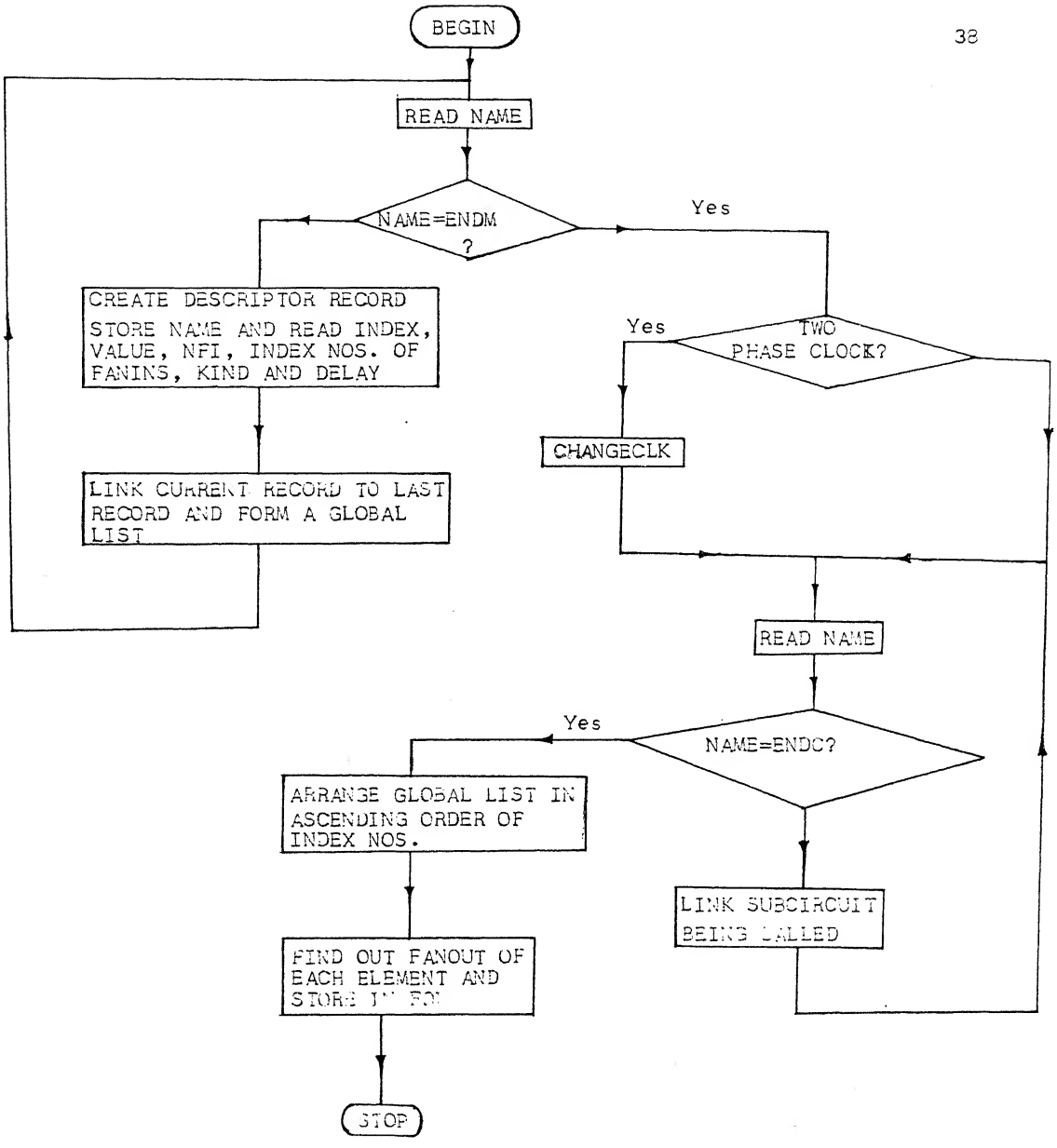


Figure 4.2(a) READ TABLE

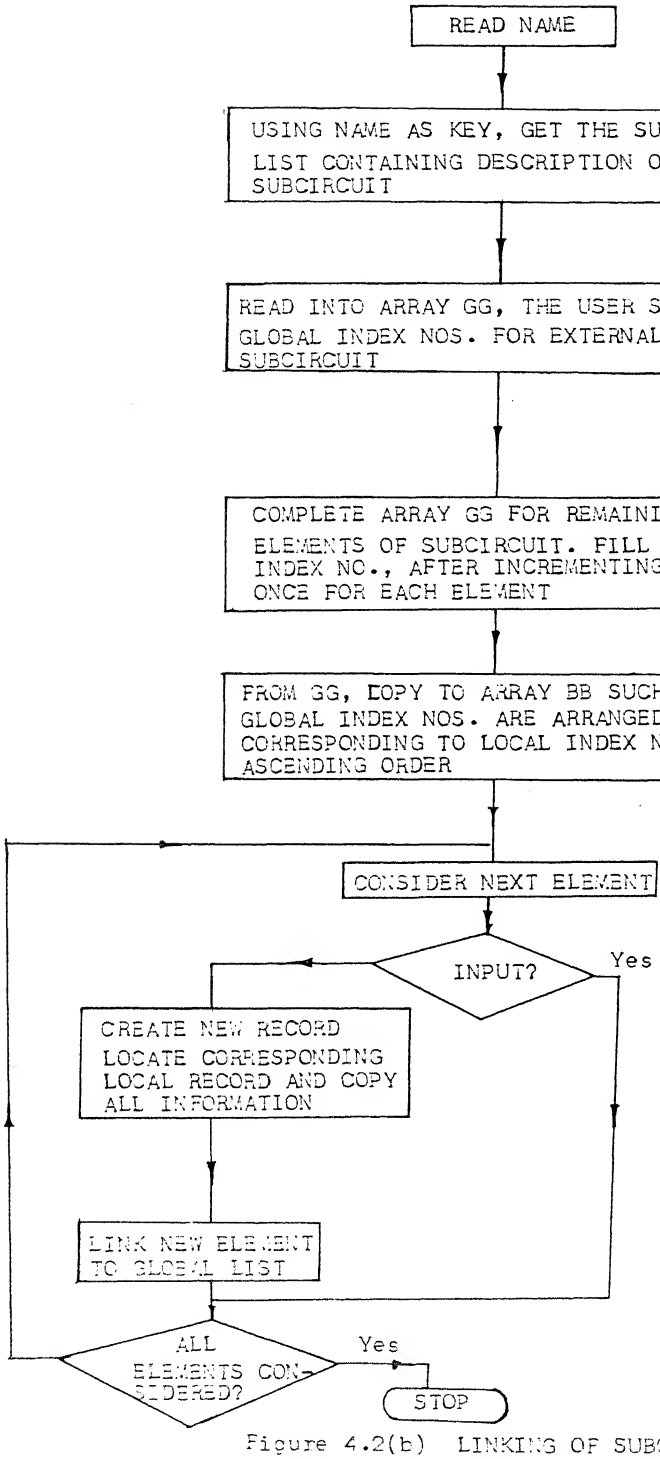


Figure 4.2(b) LINKING OF SUBCIRCUIT

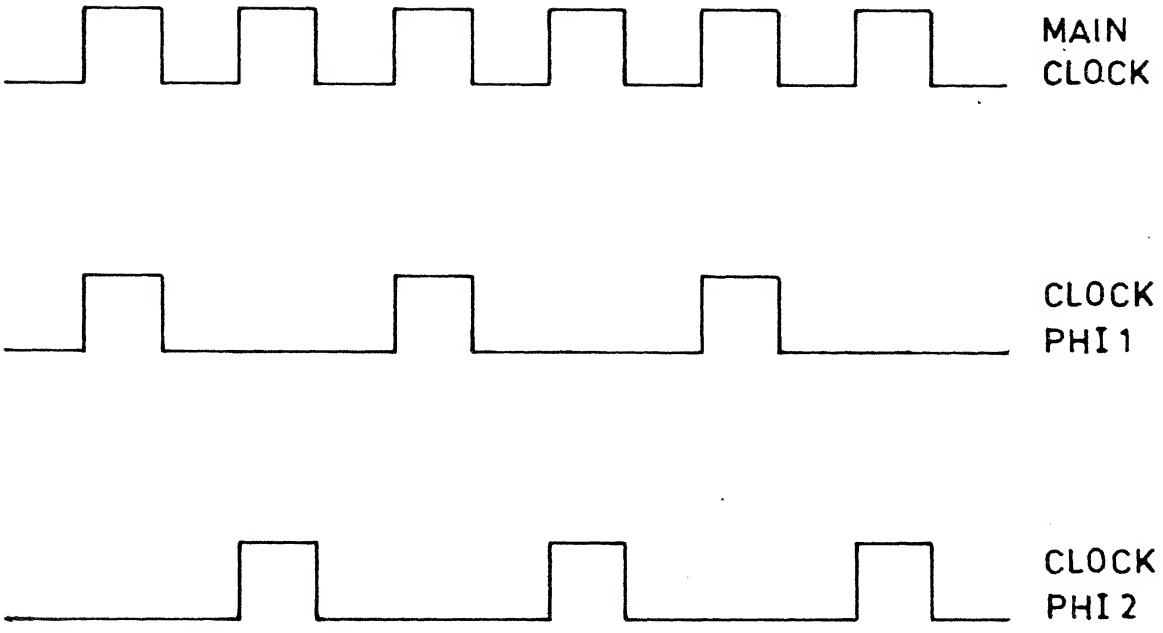


FIGURE 4.3a. TIMING SHOWING PHI1 AND PHI2.

4.2.2 Procedure CHANGECLK

Whenever this procedure is called, the logic value of clock will change either from 0 to 1 or 1 to 0. Similarly it also monitors the values of PHI1 and PHI2 clocks when they are used in any circuit.

Finally this procedure finds out fanouts of main clock and of PHI1, PHI2 if their value changes and places them in new potentially active element list. The flow chart of this procedure is given in Figure 4.3(b).

4.3 INITIAL ANALYSIS

PROCEDURE INITIALANAL : This procedure calculates the values of all elements at time t = 0. It will be evaluating each element until all the elements will have a known value 0,1 or 3. But if some element continues to have unknown value 2, even after it is evaluated certain number of times, then the procedure will be terminated and the values of all elements at that time will be returned from the procedure.

4.4 SCHEDULER MODULE

PROCEDURE SCHEDULER : This procedure schedules the elements whose value should be updated at the time when it is called. The flow chart is given in Figure 4.4 which is self explanatory.

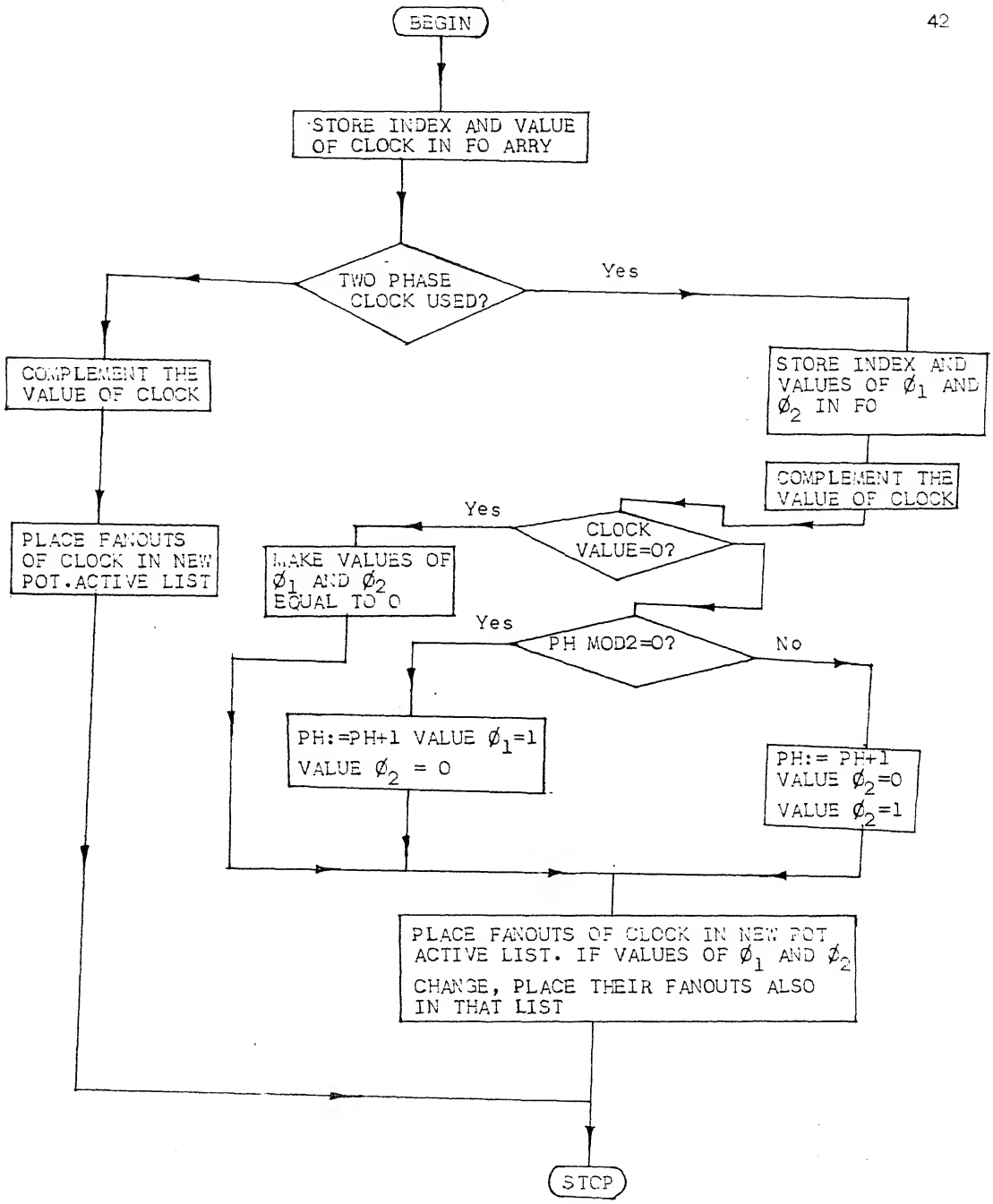


Figure 4.3(b) CHANGECLK FLOWCHART

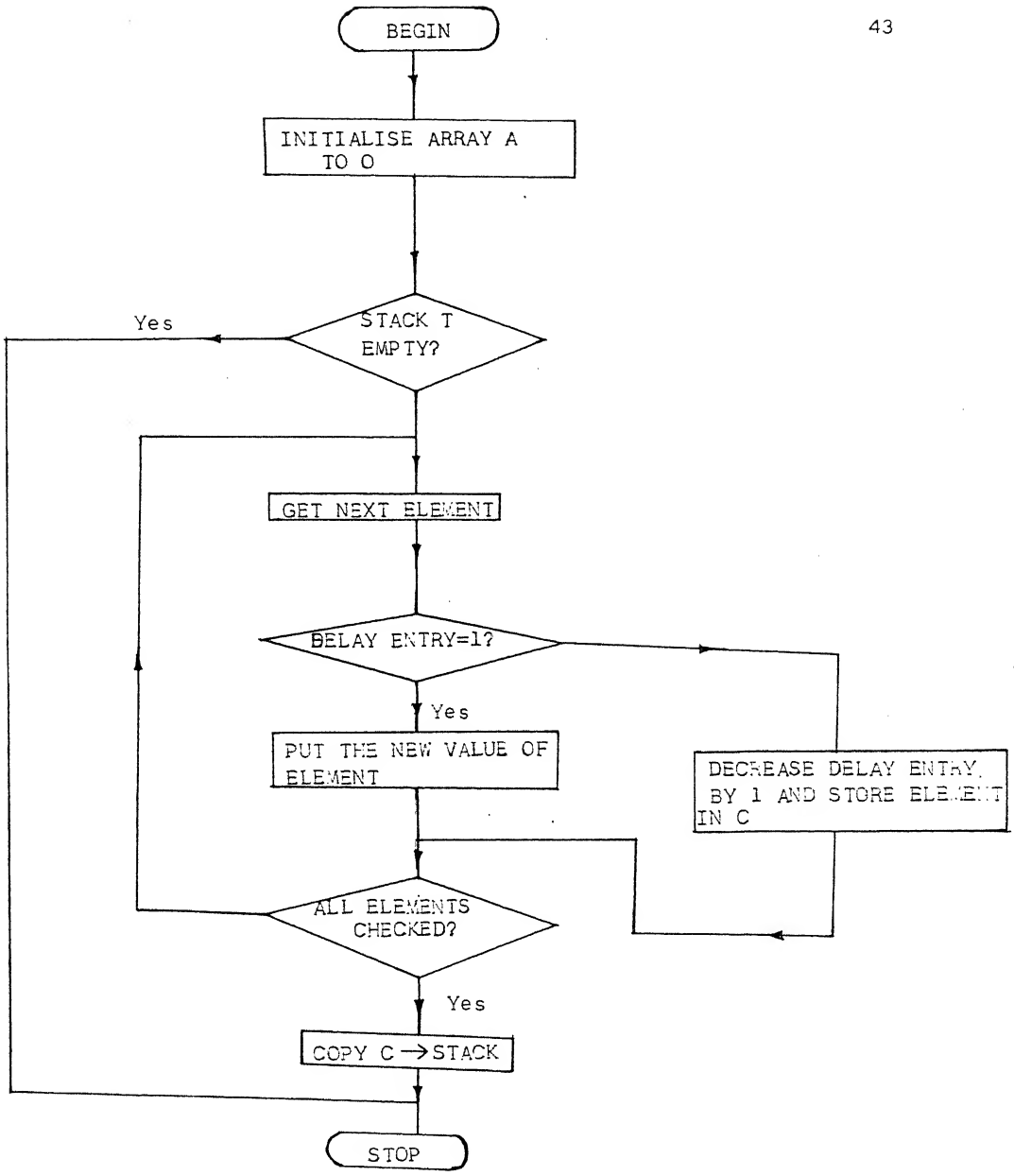


Figure 4.4 SCHEDULER FLOW CHART

4.5 EVALUATE MODULE

4.5.1 Evaluation of Gates

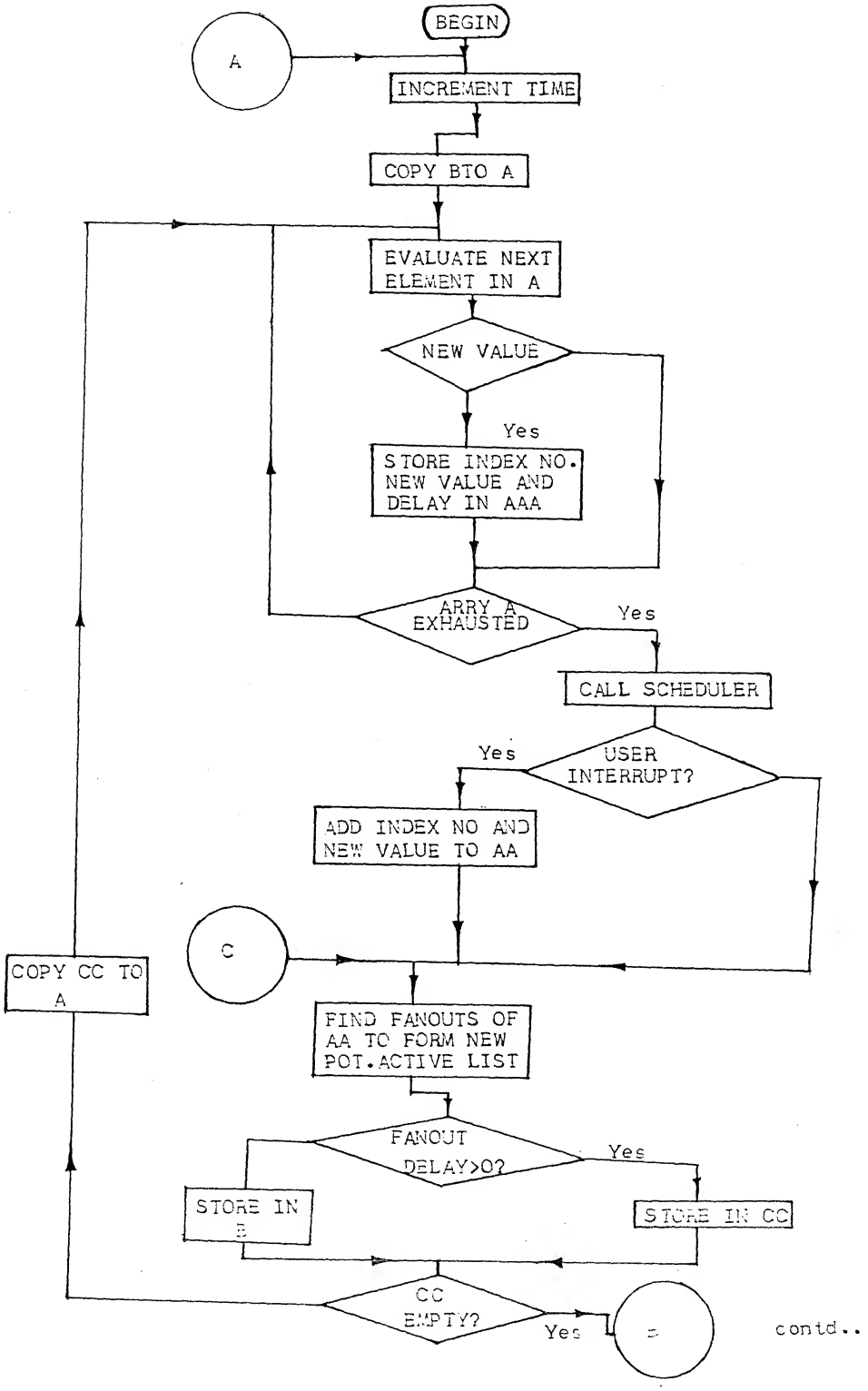
This consists of procedures EVABUS, EVAAND, EVANOT, EVAXOR, EVAXNOR, EVANAND, EVANOR, EVAOR, EVABUFF and RATIONODE where the truth table evaluation is carried out, i.e., they determine output of logic gates given the values of their inputs. Here the effect of dominant input is also taken into account to speed up evaluation, e.g., a 0 input to an AND gate will result in 0 output.

This also includes evaluation of transmission gates, evaluation of Bus elements and evaluation of Rationode logic.

4.5.2 Procedure EVALUATE

This procedure is the centre of entire program and carries out more or less all the steps shown in flow chart 2.1. But there are some additions to it such as differentiating synchronous and combinational circuits, clock transitions, printing of results and stopping criteria. The modified flow-chart, as implemented in SIMLOG is shown in Figure 4.5.

Also simulation time jumps to next clock transition time, whenever activity ceases to exist in case of synchronous circuits. This is a time saving feature of SIMLOG. However, if during skipped time, a user desired interrupt exists, simulation time is setback to the desired time.


A
BEGIN
INCREMENT TIME
COPY BTO A
EVALUATE NEXT ELEMENT IN A
NEW VALUE
Yes
STORE INDEX NO. NEW VALUE AND DELAY IN AAA
ARRY A EXHAUSTED
Yes
CALL SCHEDULER
Yes
USER INTERRUPT?
ADD INDEX NO AND NEW VALUE TO AA
C
COPY CC TO A
FIND FANOUTS OF AA TO FORM NEW POT.ACTIVE LIST
FANOUT DELAY>0?
Yes
STORE IN B
STORE IN CC
CC EMPTY?
Yes
B

contd..

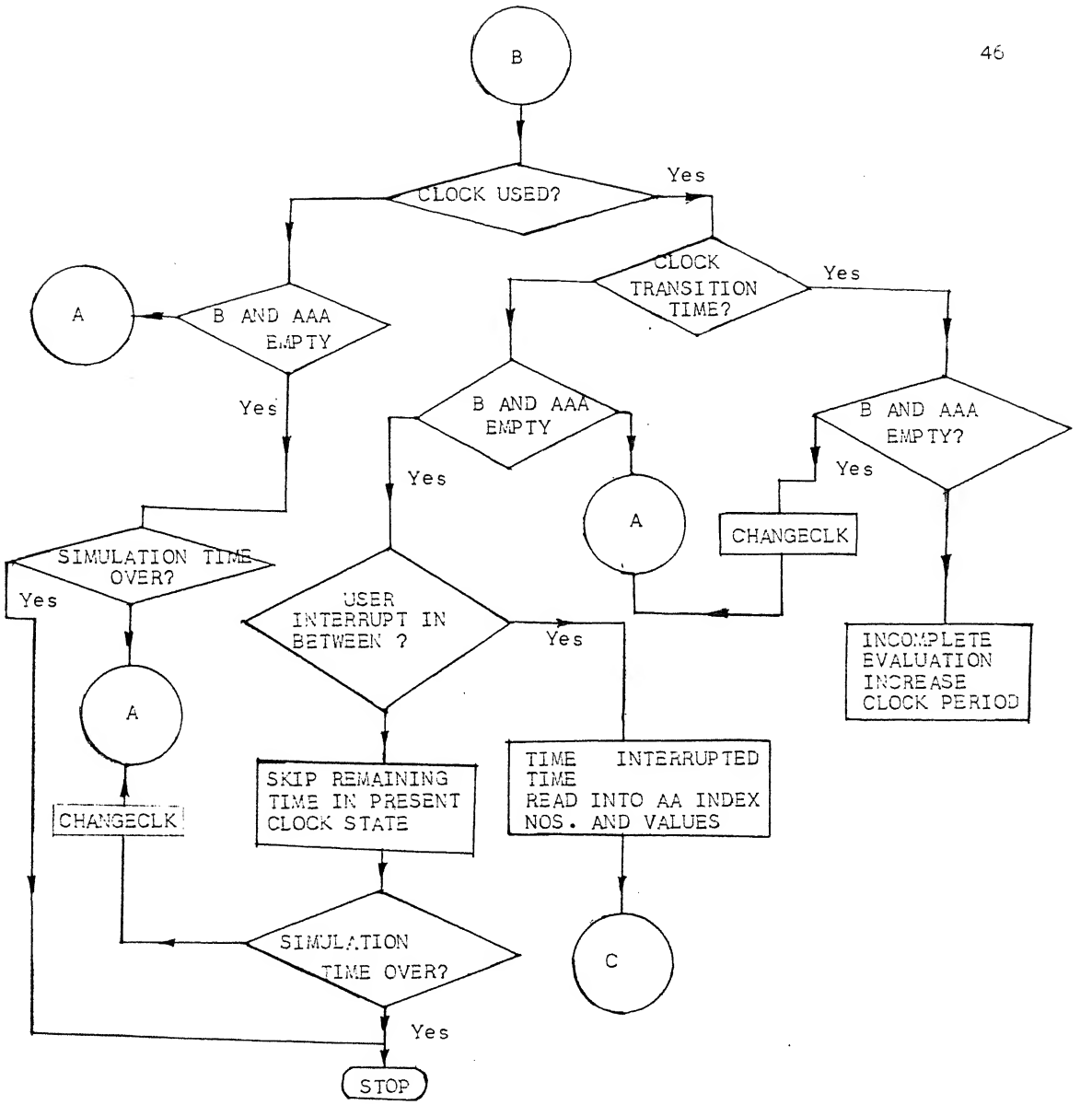


Figure 4.5 EVALUATE FLOW CHART

4.6 OUTPUT MODULE

PRINTTABLE : This prints the interconnection of different elements of a circuit in a table form giving fanins and fanouts of each element. It also gives other informations such as initial value, delay and kind or type of element.

PRINTRES : This prints the elements that constituted the potentially active list, at the time it is called with their updated values. Also values of elements being monitored will be printed along with each of potentially active element.

PRINTLINE : Prints only once, the elements being monitored along with their new values at any given time, if there is atleast one potentially active element, at that time. Also a change of state, in the logic value of those elements being monitored, is indicated by dashes —— .. . Thus a waveform like output is obtained.

4.7 MAIN PROGRAM

The general flow chart of main program is given in Figure 4.6. When a sequential circuit is simulated, Index number of clock is stored and passed onto changeclk procedure, whenever it is called. Initially total number of elements with unknown values is found and passed onto procedure INITIALANAL. One of the arrays from INITIALANAL is passed onto EVALUATE procedure.

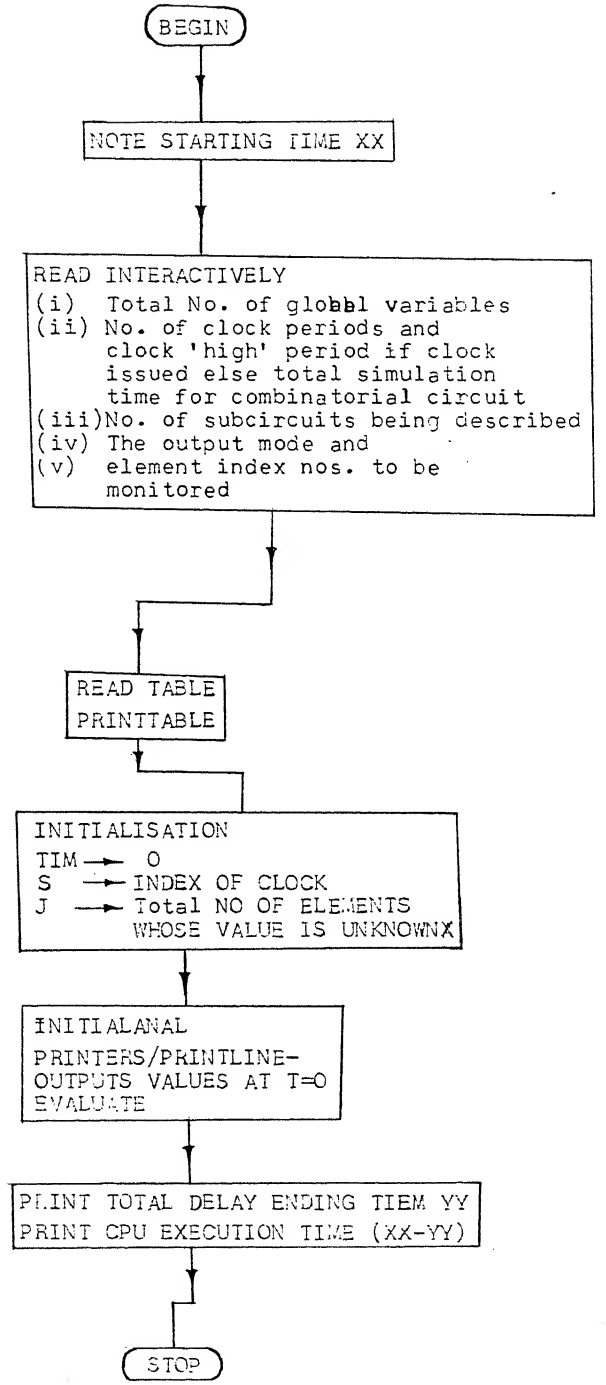


Figure 4.6 FLOW CHART OF MAIN PROGRAM

Finally the total CPU time for execution of entire program is found out and printed along with total time steps or delay.

CHAPTER 5

SIMULATION OF SOME SIMPLE CIRCUITS

Different circuits have been simulated using SIMLOG and the results are compiled in this chapter. The design and simulation of a NMOS microprocessor bit slicer is given in Chapter 6. The simulation time is also given which helps, at least in one case, to compare the time taken by timing simulator, MOSSIMR [16].

The plots are of two kinds. One gives print out of all elements that formed a potentially active list, along with their new values. The other gives the print out of the values of the elements being monitored. Results of circuit with arbitrary delay are also included which can be used to compare the results of same circuit having unit delay elements.

5.1 ONE BIT FULL ADDER

This is a typical combinatorial NMOS circuit and output response can be studied by monitoring sum and CARRY outputs. The circuit used is given in Figure 5.1 [16].

The description of input is given in listing 5.1. There are totally 19 elements in the circuit and each element is assigned an index number. Total, the number of global elements described is 19 and subcircuits are not described. Simulation is carried out for 120 time steps. Index number of sum is 16 and that of Carry is 15.

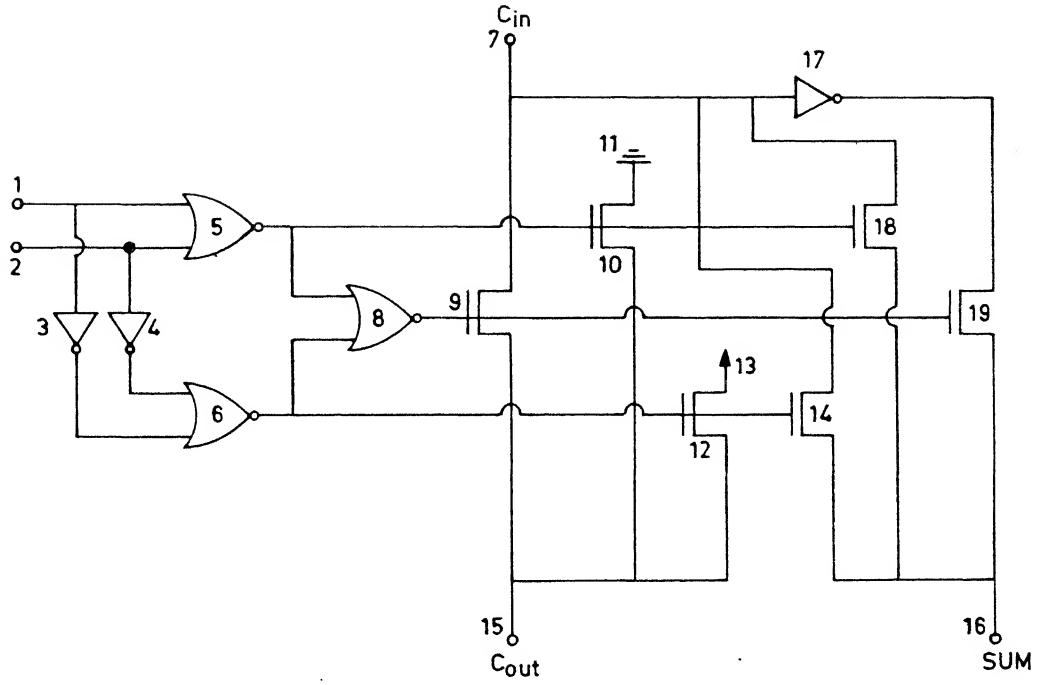


FIGURE 5.1. ONE BIT FULL ADDER : SIMLOG DESCRIPTION.

Figure 5.2 shows the output obtained by simulating one bit full adder, with all elements having a unity delay. Initially input A (Index No.1) is 1, input B (2) is 0 and input C in (7) is 1. So output sum (16) is 0 and Carry (15) is 1.

At time step 10, input 7 has become 0, forcing 16 to become 1 and 15 to 0 at time step 12. At time step 30, input 1 has become 0 and 2 has become 1 and so there is no change in outputs, 15 and 16. At time step 50, input 7 has become 1, which alters output 15 to 1 and output 16 to 0 at time step 52. At time step 70, input 1 has become 1 (i.e., all inputs are 1) which forces both sum and Carry to 1, at time step 74. At time step 90, all 3 inputs become 0, making sum and Carry also 0 at time step 92. Figure 5.2(a) shows values of all elements at time t = 0 and also shows updated values of all potentially active elements at each time step. If there are no potentially active elements at any time step, then that time step will be skipped. Figure 5.2(b) shows the waveform like output for same circuit. This type gives values of monitored elements, when at least one potentially active element occurs, at any given time.

The figure 5.3 represents the output obtained by simulating the above circuit with all elements having a delay of 5 time steps. The output values due to different inputs are obtained as explained above but they are delayed accordingly.

The total execution time in these cases is about 1.4 to 1.5 seconds. The same circuit when simulated using MOSSIMR, took about 1.91 seconds for one set of input values. SPICE took about 18.5 seconds to simulate the same circuit, for one set of input values.

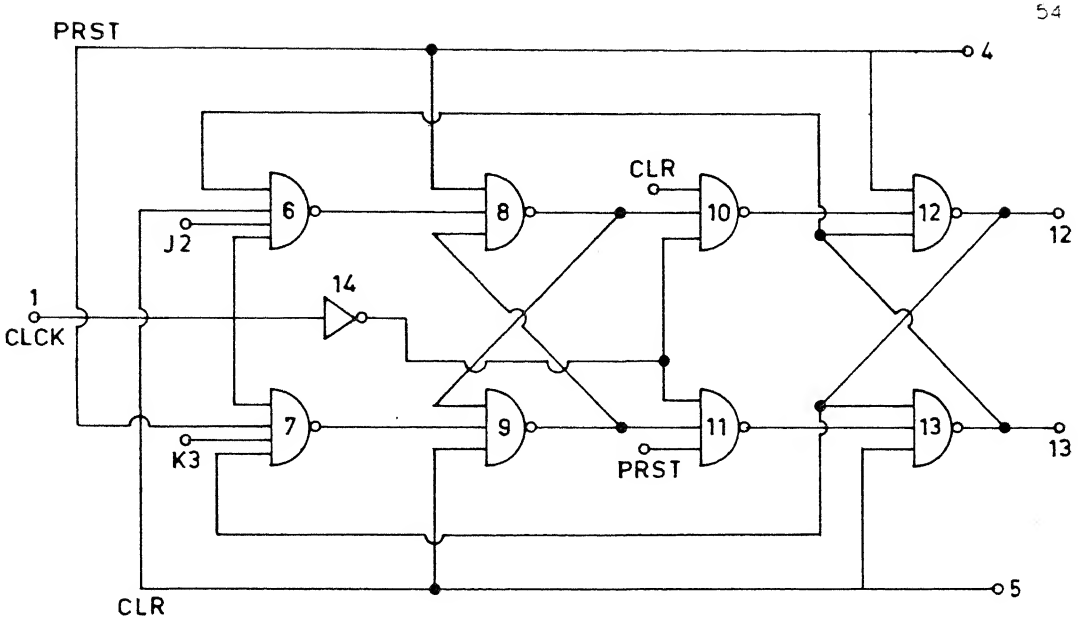
5.2 COUNTERS

5.2.1 Ripple Counter

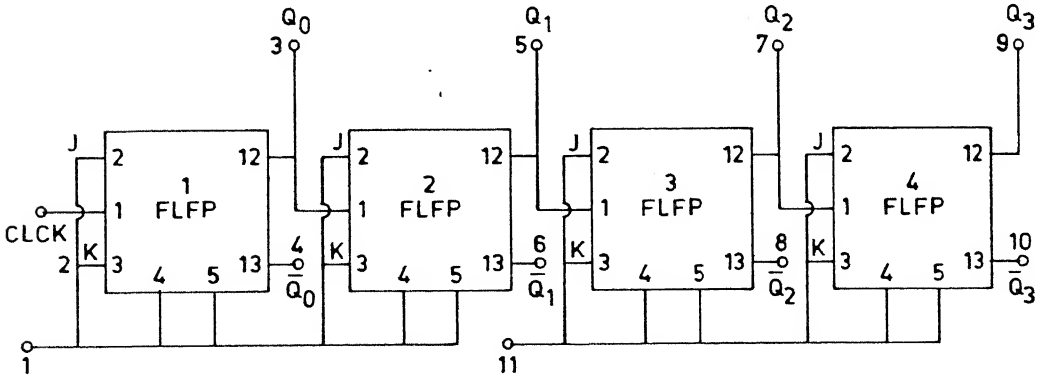
This is a standard binary ripple counter having 4 J-K master slave flip flops. J-K M-S flip flip is defined as a subcircuit and this subcircuit is called 4 times, as shown in listing 5.2. The figure is shown in 5.4.

In this case, a clock has been used. The clock high time is 25 time steps so clock period is 50 time steps. This is inputted interactively. Figure 5.5 shows simulation results obtained when all the elements of counter have a unit delay. But, to obtain the simulation results of Figure 5.6, the delays of various elements, were made proportional to their fanins.

There are totally 39 elements. But only 11 elements are globally numbered initially, other elements will get their global numbers through subcircuits calls. The four outputs are divide by 2 (Index No.3), divide by 4 (5), divide by 8 (7) and divide by 16 (9).



(a) J-K FLIPFLOP SUB-CIRCUIT.



(b) MAIN CIRCUIT

FIGURE 5.4. RIPPLE COUNTER.

In the unit delay case, a change introduced at clock input, induces a change at LSB (3) after 2 or 3 time steps and at MSB (9) after 13 to 14 time steps. But in arbitrary case delay case, it takes more time. When change occurs from 0111 to 1000, states like 0110, 0100 and 0000 are introduced in between. These can erroneously latch a data or decode. This is a drawback of the ripple counter which has been depicted by the simulator.

5.2.2 Synchronous Counter

The input is given in listing 5.3. Here results are given with elements having delay proportional to their number of fanin (Figure 5.8). The circuit connections are shown in Figure 5.7. In this case, the changes in values of LSB to MSB occur simultaneously. This is because each flip flop is clocked using same clock and there is no ripple effect. In this case both LSB and MSB, take same number of time steps, which depends upon delays of various elements of the circuit, to follow a clock change.

5.3 4 BIT ADDER/SUBTRACTOR

The input is given by listing 5.4. The circuit used is shown in Figure 5.9. When input S is 0, the circuit performs A+B. When S is equal to 1, the circuit performs A-B by taking 2's complement of B [9]. For this, each B input requires an exclusive OR gate. There are totally 77 elements. Only 21 elements are numbered globally, to start with. Remaining elements

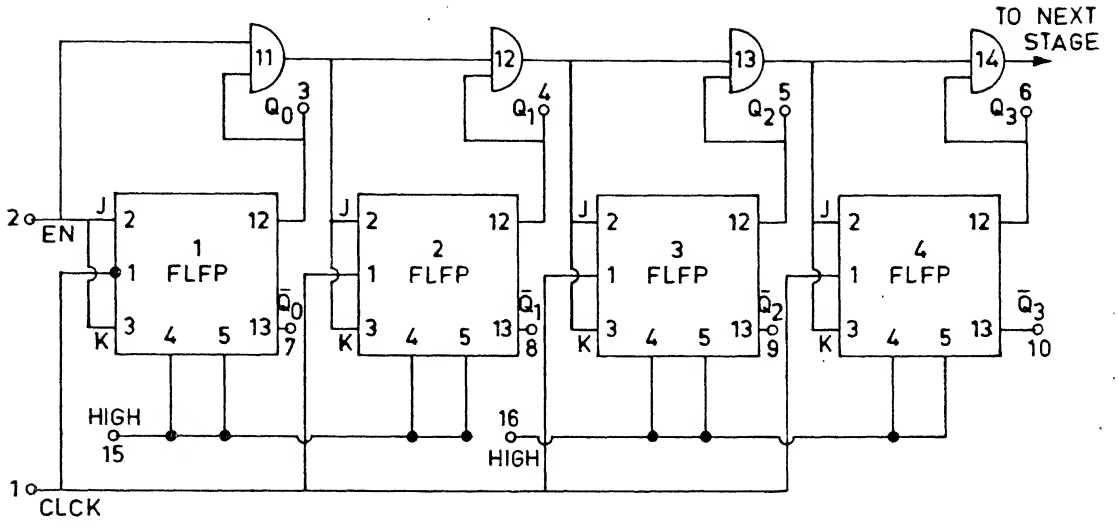


FIGURE 5.7. SYNCHRONOUS COUNTER.

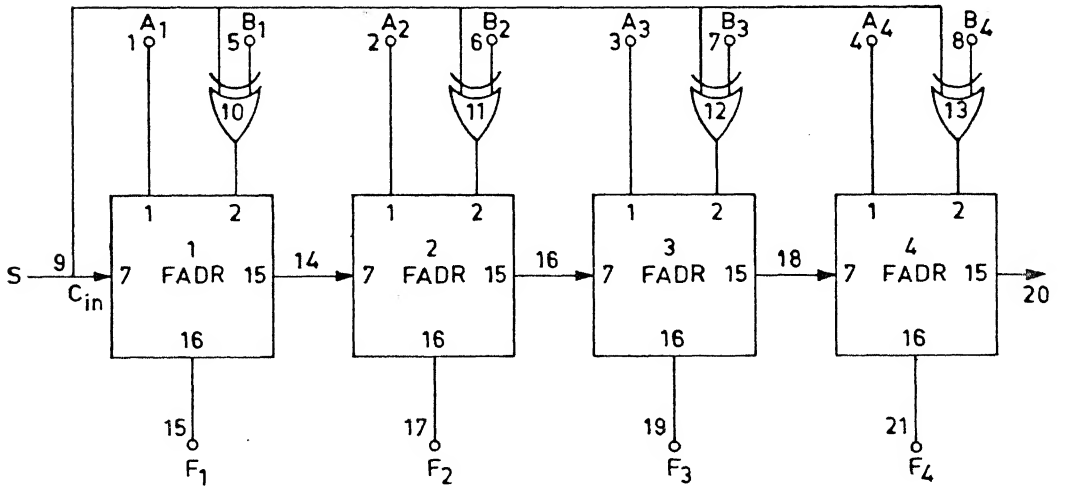


FIGURE 5.9. 4 BIT ADDER/SUBTRACTOR.

are given global numbers when subcircuit calls occur.

Simulation results are shown in Figure 5.10. In this case also, the delays of elements are proportional to their NFIs. Till time step t equal to 10, S which is C_{in} (9) is 0. So addition is performed on operands 1000 and 0011 producing result 1011. At time step 10, S becomes 1 and operands remain same. So subtraction of 0011 from 1000 takes place, producing the result 0101. At time step 70, S is changed to 0 and A operand is changed to 1011. So A+B is performed producing the result 1110. At time step 130, S is changed to 1 and A to 1010. So the result 0111 which is nothing but A-B is produced. All results are produced after some delay.

CHAPTER 6

DESIGN AND SIMULATION OF NMOS BIT SLICER

In this chapter the design of a four bit NMOS microprocessor slicer is explained. At the end, the results from the simulation of chip are also given. This chip is a cascadable chip, designed partly on the lines of a bipolar microprocessor chip AM 2903 [Ref. 7,8], and partly based on data path design given in [2]. This is designed for use in CPU's, peripheral controllers and other applications. The current instruction facilities for performing different operations, can be very useful. This can also be improved by including more number of control bits, in addition to the existing four bits. This naturally increases the range of functions that can be performed. Right now it can perform 16 operations.

6.1 CHIP DESCRIPTION

All data paths are 4 bits wide. As shown in the block diagram (Figure 6.1), the device consists of a 16 word by 4 bit, two port RAM, a (high performance) ALU and shifter. The connection diagram is shown in Figure 6.2.

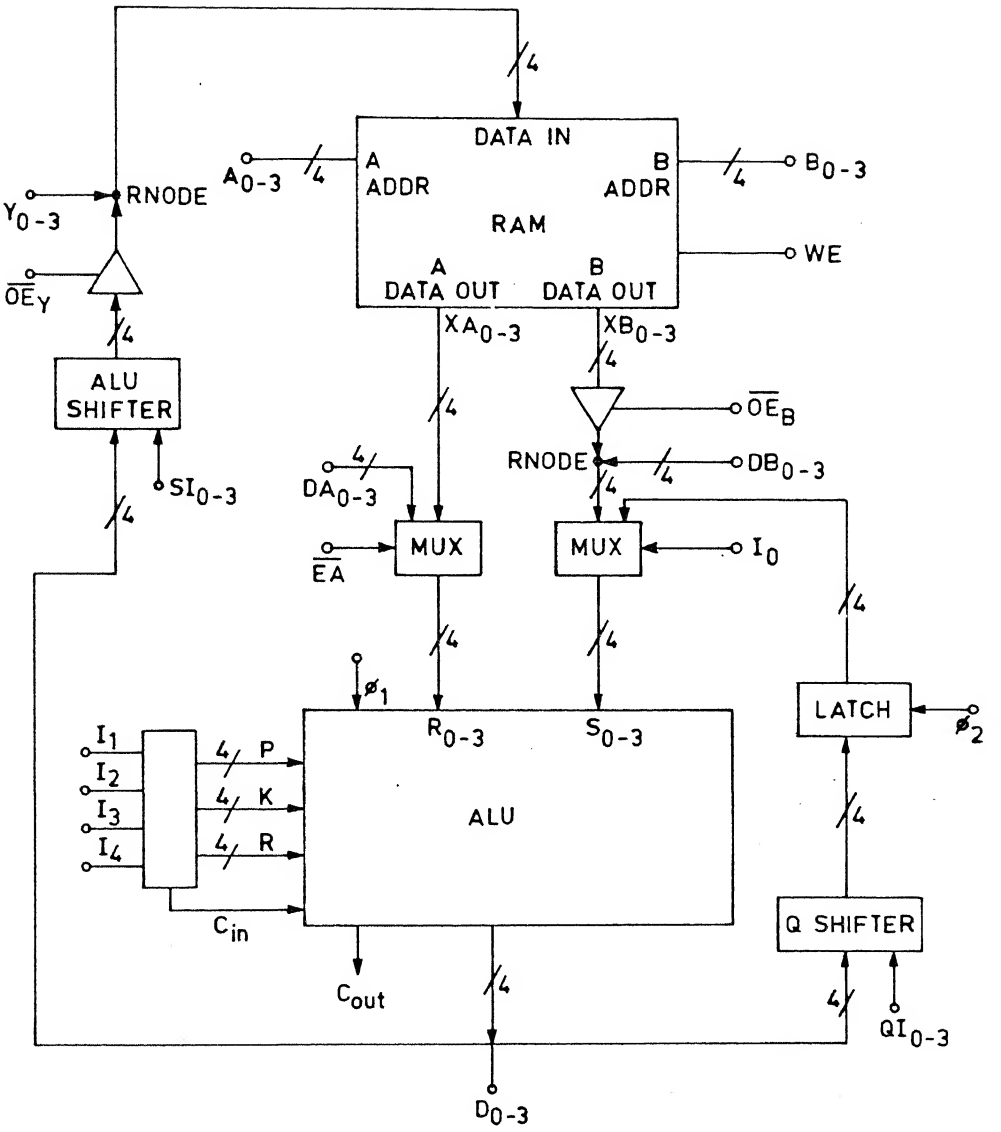


FIGURE 6.1. BLOCK DIAGRAM.

A_{0-3} : 4 RAM address inputs which contain the address of the RAM word appearing at the RAM A output port.

B_{0-3} : 4 RAM address inputs which contain the address of the RAM word appearing at the RAM B output port and to which new data is written when WE input is high.

WE : RAM write enable input. If WE is high, data at the Y port is written into RAM. When WE is low, writing into RAM is inhibited.

DA_{0-3} : A four bit external input which can be selected as one of the ALU operands at R input DA_0 is LSB and DA_3 is MSB.

DB_{0-3} : A four bit external input under control of $\overline{OE}_B$ input, data on these lines can be selected as operand of ALU.

$\overline{OE}_B$: A control input which when low disables RAM output port B and enables DB_{0-3} inputs. But when it is high, RAM output port B overrides DB_{0-3} because of 'RNODE' Logic.

$\overline{EA}$: A control input which, when HIGH, selects DA_{0-3} as ALU R operand and when low, selects RAM output port A as ALU R operand.

I_{0-4} : I_0 is a control bit, which when HIGH allows Q latch outputs to be the ALU S operand and when low, allows either DB_{0-3} or RAM output port B depending on $\overline{OE}_B$, to be ALU S operand.

I_{1_4} are 4 control bits which decide the function to be performed by chip.

SI_{0-3} : Four inputs controlling the ALU shifter. Depending on these bits, multibit shift operations are possible.

QI_{0-3} : Four inputs which control the Q shifter.

Y_{0-3} : 4 data inputs, under control of $\overline{OE}_Y$ which when low allows these bits to dominate and act as input to RAM which will be written into RAM.

$\overline{OE}_Y$: A control input, which when HIGH enables ALU shifter output to be directly connected to RAM data overriding Y_{0-3}. But when low, this allows Y_{0-3} to be connected to RAM.

C_{OUT} : This output indicates the carryout of ALU of bit slicer.

D_{0-3} : These are 4 output bits of the chip. These are connected out from ALU.

The description of different blocks now follows :

6.2 RAM MODULE

Any two RAM addresses at the A and B address ports can be read simultaneously, at the respective A and B output ports. Identical data appear at the two output ports when same address is applied to both address ports.

External data at Y port can be directly written into RAM or data at ALU shifter output can be entered into RAM, only if WE is HIGH.

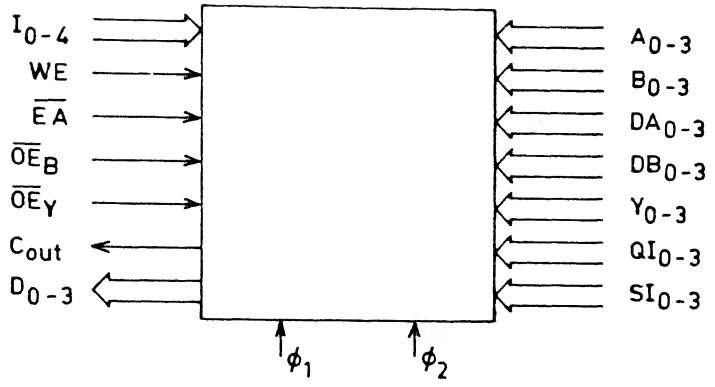


FIGURE 6.2. CONNECTION DIAGRAM.

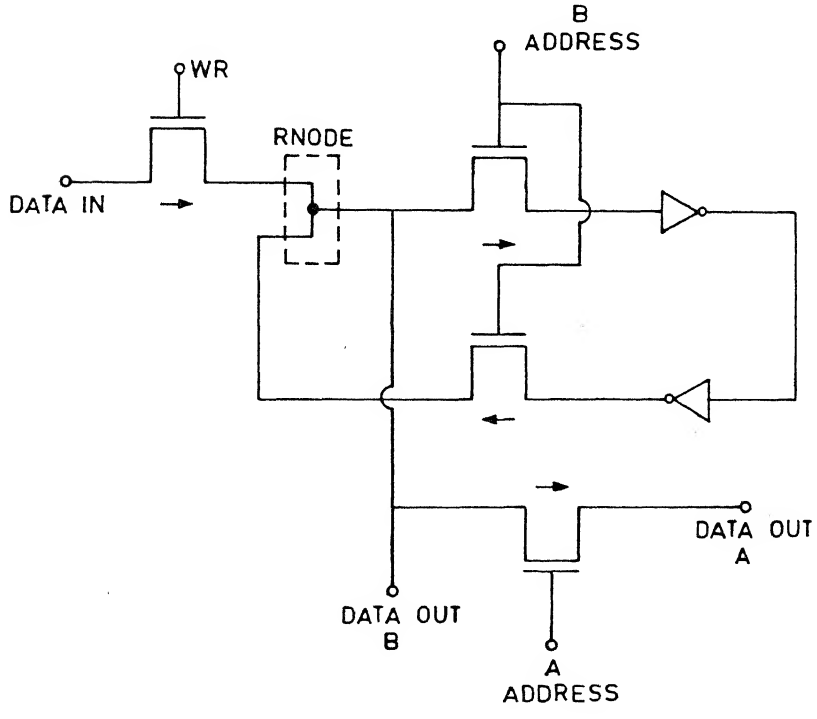


FIGURE 6.3. RAM CELL.

WR of Figure 6.3 is connected from an AND gate, the inputs to which are WE and corresponding output of B address decoder. This will enable the writing into RAM, when WE is HIGH and also ensures that only address at B address port to be the destination address for writing data.

6.3 ARITHMETIC LOGIC UNIT MODULE

The ALU is capable of performing a large number functions, the reason being there are 12 control pins (K_{0-3} P_{0-3} and R_{0-3}) as shown in block diagram [2]. But for simplicity it has been designed to perform only 16 operations. This is achieved by using a logic circuit which maps 4 instruction control inputs I_{1-4} to 12 control inputs and one C_{in} input. I_{1-4} give 16 different combinations of 13 K,P,R and C_{in} inputs.

The functions that are performed by the ALU have been tabulated in 6.4. As already mentioned, this logic circuit can be changed to allow a larger function table by using more number of instruction control inputs. Obviously K,P and R inputs can take 2^{12} values. But some of the combinations may produce identical functions.

The two basic blocks of the ALU are the Carry Chain unit and multiplexer unit. They are depicted in Figures 6.5 and 6.6 respectively.

I_3	I_2	I_1	K	P	R	C_{IN}	FUNCTION
0	0	0	0	0	0	0	Zero
0	0	1	4	9	6	1	S - R
0	1	0	2	9	6	1	R - S
0	1	1	1	6	6	0	R + S
1	0	0	5	10	6	1	S + 1
1	0	1	0	5	12	0	NOTSS
1	1	0	3	12	6	1	R + 1
1	1	1	0	3	12	0	NOT R
0	0	0	0	12	12	0	R
0	0	1	0	10	12	0	S
0	1	0	0	14	12	0	A/O
0	1	1	0	6	12	0	R EXOR S
1	0	0	12	3	9	1	R - 1
1	0	1	0	8	12	0	R AND S
1	1	0	10	5	9	1	S - 1
1	1	1	0	14	12	0	R OR S

Figure 6.4 FUNCTION TABLE

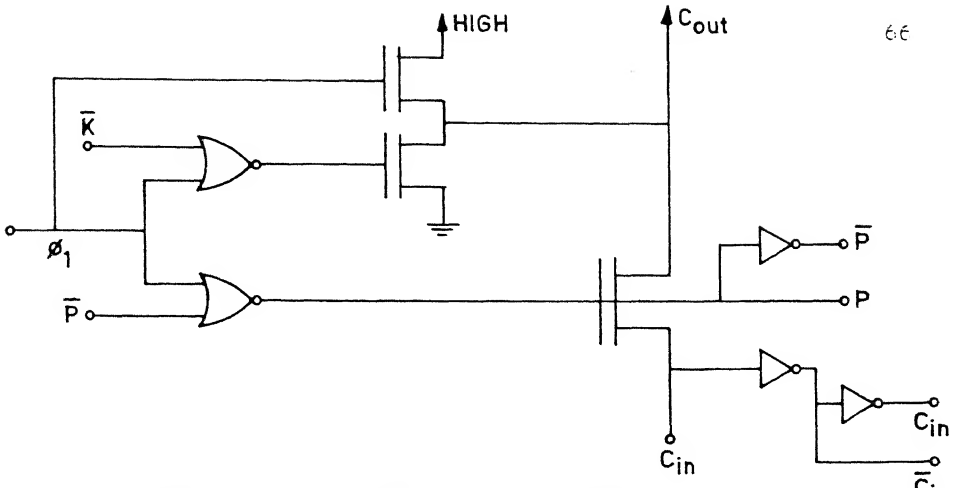


FIGURE 6.5. CARRY CHAIN CIRCUIT.

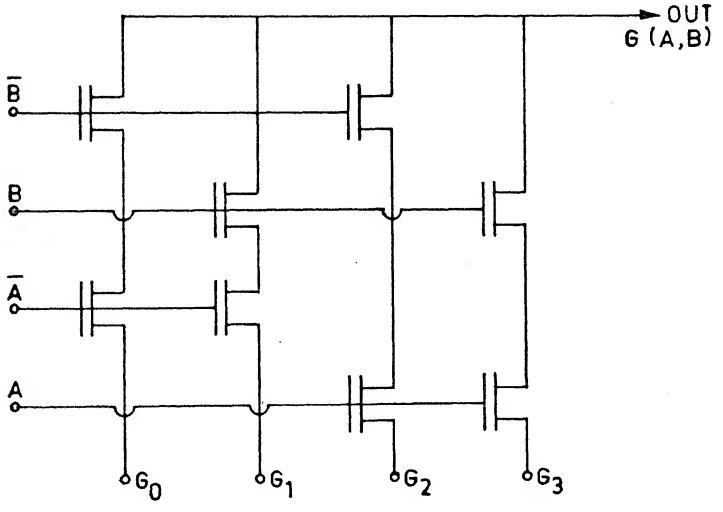


FIGURE 6.6. FUNCTIONAL BLOCK.

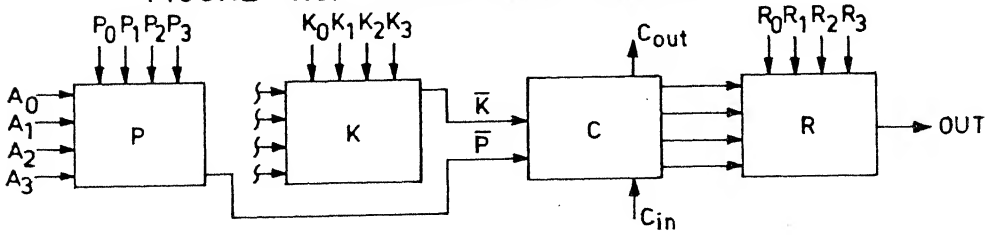


FIGURE 6.7. ONE BIT ALU.

In the ALU there will be a null period before any operation has to be performed, during which time, the precharging of Carry Chain takes place. Rise transient would be particularly slow whenever high signals have to propagate through a pass transistor. So precharging of Carry Chain is done, in order to avoid the propogation of high signals through pass transistors. The clock will be 'HIGH' during this period which disables the NOR gates in figure. The Carry Chain unit can propagate a Carry, generate a Carry or can kill a Carry, depending on $\overline{K}$ and $\overline{P}$ inputs to NOR gates.

The multiplexer unit is another functional block. It has 4 control inputs and two input operands. The block also needs inverted values of input operands. It can implement 16 logic functions depending on control inputs. The set of transistors, connects one and only one of the vertical control lines to the output, depending on its input combination.

Figure 6.7 gives connections for a one bit ALU. It has 3 functional blocks (K,P and R multiplexers) and one Carry chain block (C). For a 4 bit ALU, 4 such units will be connected, with C_{out} of first unit connected to C_{in} of next unit and so on. The control bits P_{0-3}, K_{0-3} and R_{0-3} remain same for all such units.

The ALU performs its functions when clock ϕ_1 is low. When it is high, as already mentioned, precharging will take place. The

input operands should be present at ALU inputs when clock ϕ_1 goes high and the result will be produced soon after the clock goes low.

6.4 SHIFTER AND OTHER MODULES

Shifter is a 4 bit by 4 bit barrel shifter and hence it is a multibit shifter. The circuit is as shown in Figure 6.8. This has 4 shift control inputs and four data inputs. The output depends on values of control inputs and at any given time one and only one input should be 'HIGH'. The number of shifts for different combinations of shift control inputs are tabulated in Table 6.9.

The other units include a latch which is a simple D latch, latching the data during clock ϕ_2, a multiplexer unit which helps in choosing different operands to R and S inputs of ALU. Here $\overline{EA}$ acts as control for R multiplexer and I_0 acts as control for S multiplexer.

6.5 SIMULATION RESULT OF BIT SLICER

The simulation of the above designed chip has been carried out and its results are also included (Figure 6.11).

6.5.1 Input Description

Totally 9 subcircuits have been described initially in the input file. The number of elements having global numbers to

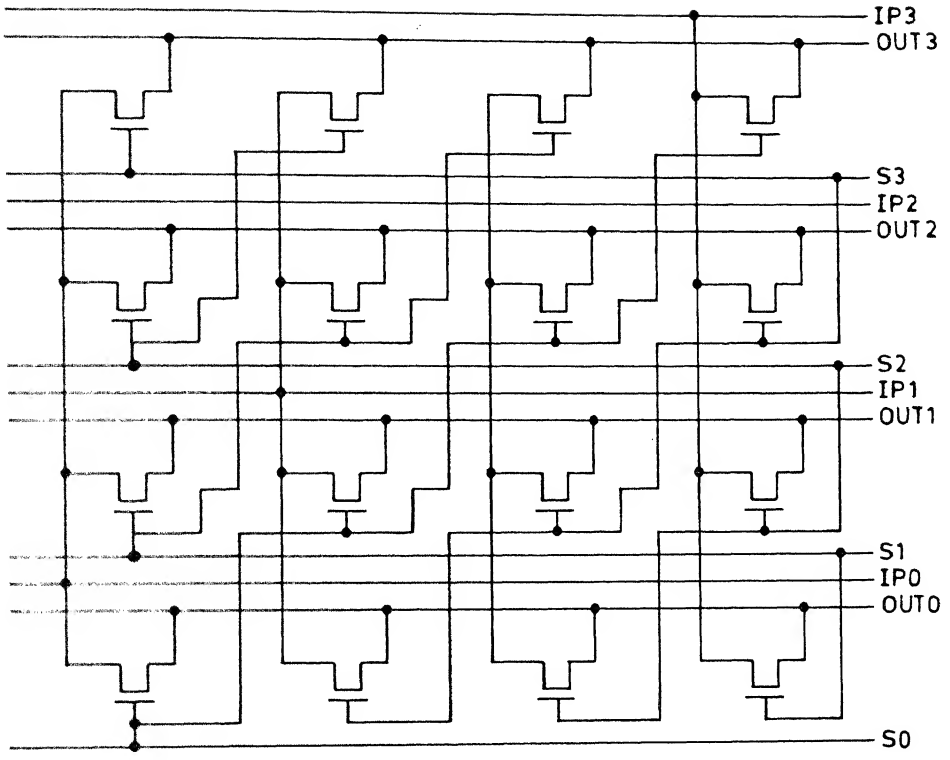


FIGURE 6.8. 4x4 BARREL SHIFTER.

S_3	S_2	S_1	S_0	No. OF BIT SHIFT (ROTATE LEFT)
0	0	0	1	0
0	0	1	0	1
0	1	0	0	2
1	0	0	0	3

FIGURE 6.9. SHIFTER TABLE.

start with is 300. Simulation has been carried out for 29 clock pulses. Global Index number of clock is 182. Two phase clock facility has been used with PHI1 having global No. 301 and PHI2 having global No. 302.

The printout includes input listing 6.11a and simulation result 6.11b. The five elements being monitored include PHI1 and four ALU outputs, 262, 254, 246 and 238. Here 262 is MSB and 238 is LSB. During initial analysis, precharging of Carry Chain has taken place. When clock ϕ_1 goes low at time 50, the ALU functions R+S, since I_{4-1} has been given values 0011. Again at time 200, when PHI1 goes high precharging takes place. Here I_{4-1} have values 0010. Hence at time 250, R-S is performed. Similarly other operations have been carried out as shown in Table 6.10.

As can be seen from listing, operands R and S have taken values from all possible sources including RAM (R-S operation), from external inputs DA and DB (R+S operation), S is also loaded from Q register to perform operation S at time step 2000.

It can also be noticed from the input listing that, data has been forced to be written into 16 RAM location till simulation time t has reached 85. At time step 85, WE is made 0, i.e., disabling the writing of data into RAM.

Time	R	S	Operation	Result
50	0110	0110	R + S	1100
250	1001	0101	R - S	0100
450	1010	1011	EXOR	0001
650	1100	0101	OR	1101
850	1101	0001	AND	0001
1050	1101	0001	NOTS	1110
1250	1101	0001	NOTR	0010
1450	0101	0001	R + 1	0110
1650	0101	0101	S + 1	0110
1850	1010	0101	R	1010
2050	0101	1010	S	1010
2250	0001	0110	S - R	0101
2450	1001	0101	R - 1	1000
2650	1001	0101	S - 1	0100
2850	1001	0101	Zero	0000

Figure 6.10 OUTPUT TABLE

Note : Figure 6.11 is included in Appendix B.

CHAPTER 7

CONCLUSION

Various simulation results obtained from SIMLOG have been explained in Chapter 5. SIMLOG is a special purpose program capable of simulating various kinds of logic circuits including NMOS and CMOS digital circuits can be simulated. The simulation of an NMOS microprocessor bit slicer has been given in Chapter 6.

7.1 ADVANTAGES OF SIMLOG

This section lists some of the features of SIMLOG.

a) Improved Speed of Simulation

Analysing circuit of 700 MOSFET complexity, in 200 time steps took about 4 cpu hours on VAX 11/780 [17]. Comparatively, the simulation of NMOS microprocessor bit slicer (as explained in Chapter 6) having greater complexity (1386 MOSFETs) took 80 to 90 seconds for 2900 time steps. SIMLOG was able to findout values of elements at each time step and even some of transient changes were detected.

The main CPU cost occurring in SIMLOG is a sort of fixed cost. That is because, CPU cost for forming the tables and

evaluating values of each element at time t = 0 (i.e., Initial analysis) is very high compared to analysis at different time steps. This has been verified by reducing the number of clock pulses to 3 in case of the bit slicer, instead of 29 used. Then the execution time was about 70-75 seconds. This clearly shows that time taken for just analysing circuits is not high and that fixed time is high.

b) Data Storage Requirement

Information about different elements are stored in a linked list. The records of the list are formed dynamically, i.e., whenever a new element is read, one record is formed. This clearly reduces the memory requirement. In fact the maximum total number of elements is limited by declaration of array of pointers which will be pointing to each element as explained in 2.4. Currently the maximum limit is 2000 elements.

c) SIMLOG can be utilised to study different aspects, such as :

i) SIMLOG has two phase clock facility where 2 nonoverlapping clocks (PHI1) and (PHI2) will be formed and element descriptor record will be filled accordingly.

ii) Using the Initial analysis of SIMLOG, one can easily find out the values at time t = 0, that is in a way, a steady state analysi

iii) Another important feature of SIMLOG is finding out maximum

clock frequency permissible in case of sequential circuits. For e.g., in case of a RIPPLE COUNTER, if TTL gates with 1 nanosecond were used, the clock high and low should be at least 15 ns each in case of unit delay elements. This is because from Figure 5.6 we can see that for change to propagate from LSB to MSB it takes 12 to 13 time steps. Thus the maximum permissable frequency is $\frac{1}{30}$ ns = 33.3 MHz. When gates have delay of 10 ns, it is about $\frac{1}{83 \times 2 \text{ ns}} \simeq 6$ MHz. In case of the unit delay elements synchronous counter, of Figure 5.10, it is about $\frac{1}{10}$ ns = 100 MHz.

If a higher frequency is used, i.e., lower time steps, the SIMLOG gives the message 'Incomplete evaluation'. So from this actually the clock frequency can be designed.

iv) It is also very easy to understand the Input format for describing circuits in SIMLOG.

v) Bidirectional and unidirectional transmission gates can be easily simulated. Hence NMOS circuits can be simulated.

7.2 LIMITATIONS IN SIMLOG AND SCOPE FOR FUTURE WORK

SIMLOG provides only one level of subcircuit desdription. So subcircuits inside subcircuits cannot be described. So SIMLOG can be made more powerful by making use of nested subcircuits, i.e., providing multiple level subcircuit nesting.

Flip flops and registers will have to be defined using subcircuits. But provision can be provided to evaluate these circuits by treating them as modules, i.e., treating them also at gate level.

Bidirectional gate is modelled using 2 unidirectional transmission gates connected back to back. This increases the number of elements. So bidirectional gate should be modelled as a single element using either a different model or by trying to modify the existing data structure to suit selective trace in bidirectional gate case also.

The bus element modelled in SIMLOG will detect if there is any 'BUS CONFLICT', i.e., if 0 and 1 are put on the bus at the same time. But SIMLOG will just say to the user that a bus conflict occurs and forces that element to 0. In actual case this may not happen always. Even the bus may take on value 1. In such cases, SIMLOG will produce wrong intermediate results. This may affect the final result also. This is one of limitations of SIMLOG.

Currently, in SIMLOG, only transport delay of elements has been taken into account for simulating a circuit with arbitrary delay elements. This can be improved by providing facilities for other kinds of delays including ambiguity delay.

REFERENCES

[1] Brever and Friedman, Diagnosis and Reliable Design of Digital Systems, Computer Science Press Inc., 1976.

[2] Carver Mead and Lynn Conway, Introduction to VLSI Systems, McGraw-Hill, 1972.

[3] Lance Glasser and Danier Dobber Phul, The Design and Analysis of VLSI Circuits, Addison-Wesley 1985.

[4] S.A. Szygenda and E.W. Thompson, 'Digital Logic Simulation in a Time based, Table driven environment - Part 1 - Design Verification', Computer, March 75, pp 24-36.

[5] M.J. Howes and D.V. Morgan (Editors), Large Scale Integratio Devices, Circuits and Systems, John Wiley and Sons, 1981.

[6] Robert M. McDermott, 'Transmission Gate modelling in an Existing Three-value Simulator', 19th Design Automation Conference, 1982, IEEE, pp 678-681.

[7] Adam Osborne, Introduction to Micro Computers, vol.2, Some Real Microprocessors.

[8] Bipolar Microprocessor Logic and Interface, AM 2900 Family, 1985 Data Book.

[9] M. Morris Mano, Digital Logic and Computer Design, Ptentice-Hall of India, 1986.

[10] P.A. Bobillier, B.C. Kahan and A.R. Probst, Simulation with GPSS and GPSS V, Prentice Hall,Inc., Englewood Cliffs, N.J. 1976.

[11] P.J. Kiviat, R. Villanueva and H.M. Markowitz, The SIMSCRIPT II, A Simulation Programming Language, Prentice Hall, Englewood Cliffs, N.J., 1968.

[12] F.J. Hill and G.K. Peterson, 'Digital System: Hardware Organisation and Design', John Wiley and Sons, 1973.

[13] 'LAMP' - Bell Systems Technical Journal - October 74, pp 1442-1475.

[14] L.W. Nagel and D.O. Pederson, SPICE, 'Simulation Program with Integrated Circuit Emphasis', Electron Res. Lab., Univ. of California, Berkeley.

[15] Maj. F. Rozario, Computer Simulation of Logic Circuits, M Tech Thesis, Dept of Electrical Engineering, IIT Kanpur, 1985.

[16] Saibal Banerjee, MOSIMR: A Timing Simulator using Signal Propagation Characteristics to Exploit Latency, M Tech Thesis Dept of Electrical Engineering, IIT Kanpur, 1985.

[17] A.R. Newton and A. Sangiovannivincentelli, 'Relaxation-based Electronic Simulation', IEEE Trans. Computer Aided Design, vol. CAD-3, No.4, Oct 1984, pp 308-331.

APPENDIX A

SIMLOG : USER'S MANUAL

SIMLOG is a logic simulator which gives signal values at different time step. This manual explains how to use SIMLOG to get the desired results. It also points out different constraints and difficulties that may be faced by user in using SIMLOG in its present version.

The input is fed to SIMLOG interactively and also through input file. The output is created in output file.

To start with, the circuit should be divided into different subcircuits, if possible. Then the elements of circuit should be numbered. This will be global index number. The elements inside subcircuit should not be globally numbered. But external elements of subcircuit including output nodes should have one global index number initially. TOTAL indicates total no. of elements globally numbered initially.

The subcircuit elements will be numbered locally. Each element of subcircuit will have a local number. This includes external nodes also. So, external nodes will have one local number which is same for subcircuit wherever used and one global number.

A.1 DATA TO BE FED INTERACTIVELY

As soon as the program is executed (using commands given at the end of Appendix), the SIMLOG will ask you to input, TOTAL, the total no. of elements globally numbered to start with. Then it will ask you, whether clock has been used in the circuit. If cloc is used, it will interactively takes period of clock 'HIGH' and number of clock pulses for which simulation is to be done. If clock is not used, then it asks for number of time steps for which simulation has to be carried out. Also if clock is used, it will enquire whether two phase clock facility is needed or not.

Then you will have to key in number of subcircuits that you are going to describe in input file and the output format (whether you need all printout of all elements forming potentially active list or just the print out of values of elements being monitored at each time step. Finally you will have to key in number of elements to be monitored, a maximum of 8 and index numbers of those elements, you wish to monitor.

Then SIMLOG will display on TTY that the formation of table is started.

A.2 INPUT FILE

The general format of input file is, you should describe all subcircuits. Each subcircuit description end is denoted by ENDS Then the description of global numbered elements should follow, the

end of which is denoted by ENDM. Subcircuit calls should be done next, the end of which is denoted by ENDC. Finally the user interruptions should be given.

The comment statements can be inserted anywhere except in user interruption part. The comment statement is denoted by a * at the beginning. The comment statements can be inserted, till the line ENDC.

A.2.1 Subcircuit Description

The first line of subcircuit description should have in order, with a blank in between, the following :

A four character name, total no. of elements in subcircuit, number of external nodes, number of input external nodes and local index numbers of external nodes.

The only condition here is that the input local numbers should be specified first while specifying external local node numbers.

From second line onwards the description of elements should be done. Here the description of Input (local) nodes is optional This is because input to all subcircuits will be destribed in either in main description or they form output nodes of some other subcircuits. The description method is exactly same as tha for main circuit element description given later.

The end of subcircuit is denoted by ENDS in last line of subcircuit description. The total number of subcircuit descripti

should be equal to the number that is inputted interactively.

NOTE : If there is not even a single subcircuit to be described then start with main description.

A.2.2 Main Description

In this all basic input elements and other elements which have been globally numbered will be described. Of course, the numbered elements which are outputs of any subcircuit should NOT be described here.

A.2.2.1 Element Description

For each element the following should be described in same order. A four character name, global index number, initial value, no. of fan ins NFI, global numbers of fanins, type or kind of the element and delay of element. Each of these things are separated by a blank.

i) Global index number is already given to each element.

ii) Name can have any alphanumeric characters. But clock input should have 'CLCK' as its name.

iii) Initial value can be 0,1,2 or 3.

iv) No. of global numbers forming fains should be equal to NFI of that element. Here global numbers of elements forming output of subcircuits should be used if they are fanins to any element.

v) Kind of Type :

The different kinds of gates/elements that can be simulated are given below along with 4 characters that should be used to specify that particular element.

ELEMENT	KIND
BASIC Input	'BSIC'
NOT Gate	'NOT'
AND Gate	'AND'
OR Gate	'OR'
NAND Gate	'NAND'
NOR Gate	'NOR'
XOR Gate	'XOR'
XNOR Gate	'XNOR'
BUFFER or TRANSMISSION Gate	'BFFR'
BUS Element	'BUS'
RATIO Node	'NODE'

vi) Delay of the elements can be anything from 0 onwards.

Basic inputs will have delay 0.

As already mentioned, output elements of subcircuit should not be described here. The end of description of all elements is denoted by ENDM. In case of the transmission gate, first fan in should be input and second fan in should be control input. In case of Ratio Node, first fanin should be stronger signal than second fanin. If the first character of any line is *, then that line is taken as a comment line.

A.2.3 Subcircuit Calls

After ENDM, the subcircuits will be called using the format given below. Each subcircuit ca call is done by

CALL <NAME> <Global Index No.s of external nodes>

Here Name is 4 character word and should be same as that given in describing the subcircuit being called.

Global numbers of external nodes are given in the same order corresponding to order of the local numbers given while describing subcircuit being called.

The end of subcircuit calls are denoted by ENDC in fresh lin

Note : If subcircuit call is not made even once, then place ENDC at next line to ENDM line. Here also, * at the beginning indicates a comment line.

A.2.4 User Interrupts

The user interrupts should be given in the lines following ENDC. The format is as follows :

First the time, at which user wishes to interrupt, should be given. Next the number of changes that the user wishes to make at that time should be given. This is followed by index No. and new value of each elements which the user wishes to change. All these are given in same line with a blank in between. A 0 indicates the end of input file description.

Note : If user does not make any interrupting place 0 in the line following ENDC, which shows that input file description is complete. Input descriptions for different circuits are given in listings 5.1, 5.2, 5.3, 5.4 and 6.1.

A.2.5 Output

The results of simulation is obtained in OUTPUT file, based on user's requirement. This will have initially the interconnections of the circuit in tabular form. Then, the evaluation results are printed. A bus conflict is indicated by ** in time column followed by index no. of that bus element. Total no. of time steps and cpu execution time are also printed at the end.

Some of the constraints that user should be aware are

i) Any circuit can have a maximum of 2000 elements, for it to be simulated using SIMLOG.

ii) The maximum of 50 external nodes are allowed for any subcircuit description.

iii) Total no. of elements of a subcircuit is limited to a maximum of 100.

iv) In transmission gate case, first fanin is input and second fanin is control element.

vi) Each gate can have a maximum fain of 10 and maximum fanout of 20.

vi) In Ratio Node case, first fanin is a stranger signal.

Finally the following commands should be given in order to execute SIMLOG on DEC 1090.

```
. R PASREL 25 P
* <FILE>.PAS
. LOAD <FILE>.PAS
. CORE 200P
. START
*
```

Actually small circuits donot need 200P extra core. Even 100P core may be sufficient. In fact NMOS microprocessor bit slicer of Chapter 6, containing of 969 elements required an extra core of 185P. But to be on the safer side it is better to use 200P core.

APPENDIX B

RESULTS

```
 TY NEWIP
[:31:08]
* THIS DESCRIBES THE FULL ADDER CIRCUIT.
* THERE IS NO SUBCIRCUIT DESCRIPTION.
* STARTS WITH MAIN DESCRIPTION.
* THERE ARE 19 ELEMENTS IN THIS CIRCUIT.
* THE CIRCUIT IS SIMULATED FOR 120 TIME STEPS.
INPT 1 1 0 BINP 0
INPT 2 0 0 BINP 0
INVT 3 1 1 2 NOT 1
INVT 4 0 1 1 NOT 1
NOR1 5 2 2 1 2 NOR 1
NOR2 6 2 2 3 4 NOR 1
CINB 7 1 0 BINP 0
NOR3 8 2 2 5 6 NOR 1
BUF1 9 2 2 7 8 BFFR 1
BUF2 10 2 2 11 5 BFFR 1
LOWI 11 0 0 BINP 0
BUF3 12 2 2 13 6 BFFR 1
HIGH 13 1 0 BINP 0
BUF4 14 2 2 7 6 BFFR 1
BUS1 15 2 3 9 10 12 BUS 0
BUS2 16 2 3 14 18 19 BUS 0
INVT 17 2 1 7 NOT 1
BUF5 18 2 2 7 5 BFFR 1
BUF6 19 2 2 17 8 BFFR 1
* NOS. 1,2 & 7 ARE INPUTS.
* SUM IS NO.16,CARRY IS NO.15.
ENDM
* THERE WILL BE NO SUBCIRCUIT CALLS.
* SO PLACE ENDC IN NEXT LINE.
* THERE ARE USER INTERRUPTIONS IN THIS CASE.
* INTERRUPTIONS ARE DESCRIBED AFTER ENDC.
ENDC
10 1 7 0
30 2 1 0 2 1
50 1 7 1
70 1 1 1
90 3 1 0 2 0 7 0
0
```

LISTING 5.1.

:08:14]
GIVEN TABLE IS :

INDEX	NAME	VALUE	NFO	NFI	FOL										FIL										KIND	DELAY
1	INPT	1	2	0	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	0
2	INPT	0	2	0	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	0
3	INVT	1	1	1	6	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	NOT	1
4	INVT	0	1	1	6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NOT	1
5	NOR1	2	3	2	8	10	18	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	NOR	1
6	NOR2	2	3	2	8	12	14	0	0	0	0	0	0	0	3	4	0	0	0	0	0	0	0	0	NOR	1
7	CINB	1	4	0	9	14	17	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	0
8	NOR3	2	2	2	9	19	0	0	0	0	0	0	0	0	5	6	0	0	0	0	0	0	0	0	NOR	1
9	BUF1	2	1	2	15	0	0	0	0	0	0	0	0	0	7	8	0	0	0	0	0	0	0	0	BFFR	1
10	BUF2	2	1	2	15	0	0	0	0	0	0	0	0	0	11	5	0	0	0	0	0	0	0	0	BFFR	1
11	LOWI	0	1	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	0
12	BUF3	2	1	2	15	0	0	0	0	0	0	0	0	0	13	6	0	0	0	0	0	0	0	0	BFFR	1
13	HIGH	1	1	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	0
14	BUF4	2	1	2	16	0	0	0	0	0	0	0	0	0	7	6	0	0	0	0	0	0	0	0	BFFR	1
15	BUS1	2	0	3	0	0	0	0	0	0	0	0	0	0	9	10	12	0	0	0	0	0	0	0	BUS	0
16	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	0	14	18	19	0	0	0	0	0	0	0	BUS	0
17	INVT	2	1	1	19	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	NOT	1
18	BUF5	2	1	2	16	0	0	0	0	0	0	0	0	0	7	5	0	0	0	0	0	0	0	0	BFFR	1
19	BUF6	2	1	2	16	0	0	0	0	0	0	0	0	0	17	8	0	0	0	0	0	0	0	0	BFFR	1

evaluation results

time=	element	value	GATE 16	GATE 15
0	1	(1)	0	1
0	2	(0)	0	1
0	3	(1)	0	1
0	4	(0)	0	1
0	5	(0)	0	1
0	6	(0)	0	1
0	7	(1)	0	1
0	8	(1)	0	1
0	9	(1)	0	1
0	10	(3)	0	1
0	11	(0)	0	1
0	12	(3)	0	1
0	13	(1)	0	1
0	14	(3)	0	1
0	15	(1)	0	1
0	16	(0)	0	1
0	17	(0)	0	1
0	18	(3)	0	1
0	19	(0)	0	1
11	9	(0)	0	0
11	17	(1)	0	0
11	15	(0)	0	0
12	19	(1)	1	0
12	16	(1)	1	0
31	4	(1)	1	0
31	3	(0)	1	0
51	9	(1)	1	1
51	17	(0)	1	1
51	15	(1)	1	1
52	19	(0)	0	1
52	16	(0)	0	1
71	4	(0)	0	1
72	6	(1)	0	1

```
**     16
73              8     (  0)        0        1
73             12     (  1)        0        1
73             14     (  1)        0        1
74              9     (  3)        1        1
74             19     (  3)        1        1
74             16     (  1)        1        1
91              4     (  1)        0        1
91              5     (  1)        0        1
91              3     (  1)        0        1
91             14     (  0)        0        1
91             17     (  1)        0        1
91             16     (  0)        0        1
**     15
92              6     (  0)        0        0
92             10     (  0)        0        0
92             18     (  0)        0        0
92             15     (  0)        0        0
93             12     (  3)        0        0
93             14     (  3)        0        0
```

** XX :INDICATES BUS CONFLICT AT GATE/ELEMENT NO. XX .

TOTAL TIME DELAY= 120 THE EXECUTION TIME : 1.510 SECONDS

FIGURE 5.2 (a)

5.2B
12:47]
GIVEN TABLE IS :

INDEX	NAME	VALUE	NFO	NFI	FOL									FIL									KIND	DE
1	INPT	1	2	0	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
2	INPT	0	2	0	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
3	INVT	1	1	1	6	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	NOT	
4	INVT	0	1	1	6	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	NOT	
5	NOR1	2	3	2	8	10	16	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	NOR	
6	NOR2	2	3	2	8	12	14	0	0	0	0	0	0	3	4	0	0	0	0	0	0	0	NOR	
7	CINB	1	4	0	9	14	17	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
8	NOR3	2	2	2	9	19	0	0	0	0	0	0	0	5	6	0	0	0	0	0	0	0	NOR	
9	BUF1	2	1	2	15	0	0	0	0	0	0	0	0	7	8	0	0	0	0	0	0	0	BFFR	
10	BUF2	2	1	2	15	0	0	0	0	0	0	0	0	11	5	0	0	0	0	0	0	0	BFFR	
11	LOWI	0	1	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
12	BUF3	2	1	2	15	0	0	0	0	0	0	0	0	13	6	0	0	0	0	0	0	0	BFFR	
13	HIGH	1	1	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
14	BUF4	2	1	2	16	0	0	0	0	0	0	0	0	7	6	0	0	0	0	0	0	0	BFFR	
15	BUS1	2	0	3	0	0	0	0	0	0	0	0	0	9	10	12	0	0	0	0	0	0	BUS	
16	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	14	18	19	0	0	0	0	0	0	BUS	
17	INV3	2	1	1	19	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	NOT	
18	BUF5	2	1	2	16	0	0	0	0	0	0	0	0	7	5	0	0	0	0	0	0	0	BFFR	
19	BUF6	2	1	2	16	0	0	0	0	0	0	0	0	17	8	0	0	0	0	0	0	0	BFFR	

evaluation results

```
TIME=        GATE  16          GATE  15
****         *******          *******

 0           0                        1
11           0                0--------
12           --------1        0
31                   1        0
51                   1        ---------1
52           0--------                1
71           0                        1
72           0                        1
**    16
73           0                        1
74           --------1                1
91           0--------                1
**    15
92           0                0------- -
93           0                0
```

** XX : INDICATES BUS CONFLICT AT GATE/ELEMENT NO. XX .

TOTAL TIME DELAY = 120

THE EXECUTION TIME : 1.402 SECONDS

FIGURE 5.2 (B)

5.3
:54:31]
GIVEN TABLE IS :

INDEX	NAME	VALUE	NFO	NF	I	FOL									FIL										KIND	I
1	INPT	1	2	0		5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
2	INPT	0	2	0		5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
3	INVT	1	1	1		0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	NOT	
4	INVT	0	1	1		0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NOT	
5	NOR1	2	3	2		10	18	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	NOR	
6	NOR2	2	3	2		12	14	0	0	0	0	0	0	0	3	4	0	0	0	0	0	0	0	0	NOR	
7	CINB	1	4	0		14	17	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
8	NOR3	2	2	2		19	0	0	0	0	0	0	0	0	5	6	0	0	0	0	0	0	0	0	NOR	
9	BUF1	2	1	2		0	0	0	0	0	0	0	0	0	7	8	0	0	0	0	0	0	0	0	BFFR	
10	BUF2	2	1	2		0	0	0	0	0	0	0	0	0	11	5	0	0	0	0	0	0	0	0	BFFR	
11	LOWI	0	1	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
12	BUF3	2	1	2		0	0	0	0	0	0	0	0	0	13	6	0	0	0	0	0	0	0	0	BFFR	
13	HIGH	1	1	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
14	BUF4	2	1	2		0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	BFFR	
15	BUS1	2	0	3		0	0	0	0	0	0	0	0	0	9	10	12	0	0	0	0	0	0	0	BUS	
16	BUS2	2	0	3		0	0	0	0	0	0	0	0	0	14	15	19	0	0	0	0	0	0	0	BUS	
17	INVT	2	1	1		0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	NOT	
18	BUF5	2	1	2		0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	BFFR	
19	BUF6	2	1	2		0	0	0	0	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0	BFFR	

evaluation results

```
TIME=      GATE 16      GATE 15
*****      ********     ********

  0          0              1
 15          0          0-------
 20          --------1      0
 35                  1      0
 55                  1      --------1
 60          0--------             1
 75          0                    1
 80          0                    1
**     16
 85          0                    1
 90          --------1            1
 95          0--------            1
**     15
100          0          0---------
105          0          0
```

** XX :INDICATES BUS CONFLICT AT GATE/ELEMENT NO. XX .

AL TIME DELAY= 120 THE EXECUTION TIME : 1.414 SECONDS

FIGURE 5.3.

```
 TY IPNEW
[23:58:08]
* THIS CIRCUIT DESCRIBES RIPPLE COUNTER.
FLFP 14 7 5 1 2 3 4 5 12 13
* THIS SUBCIRCUIT DESCRIBES J-K FLIPFLOP.
* THERE ARE 14 ELEMENTS IN THIS SUBCIRCUIT
* THERE ARE 7 EXT. NODES,5 OF WHICH ARE INPUT NODES.
MSTR 6 2 4 1 2 5 13 NAND 9
* 1 TO 5 ARE INPUT NODES.SO THEY NEED NOT BE DESCRIBED.
MSTR 7 2 4 1 3 4 12 NAND 9
MSTR 8 0 3 4 6 9 NAND 7
MSTR 9 1 3 5 7 8 NAND 7
SLVE 10 2 3 5 8 14 NAND 7
SLVE 11 2 3 4 9 14 NAND 7
SLVE 12 0 3 4 10 13 NAND 7
SLVE 13 1 3 5 11 12 NAND 7
NOT1 14 2 1 1 NOT 2
ENDS
* END OF FLIP FLOP DESCRIPTION.
INPT 1 1 0 BINP 0
CLCK 2 1 0 BINP 0
* INDEX NO. OF CLOCK IS 2.
INPT 11 1 0 BINP 0
* THE REMAINING ELEMENTS ARE FORMED BY SUBCIRCUIT CALLS.
ENDM
CALL FLFP 2 1 1 1 1 3 4
CALL FLFP 3 1 1 1 1 5 6
CALL FLFP 5 11 11 11 11 7 8
CALL FLFP 7 11 11 11 11 9 10
* THE FLIP FLOP IS CALLED 4 TIMES.
ENDC
0
```

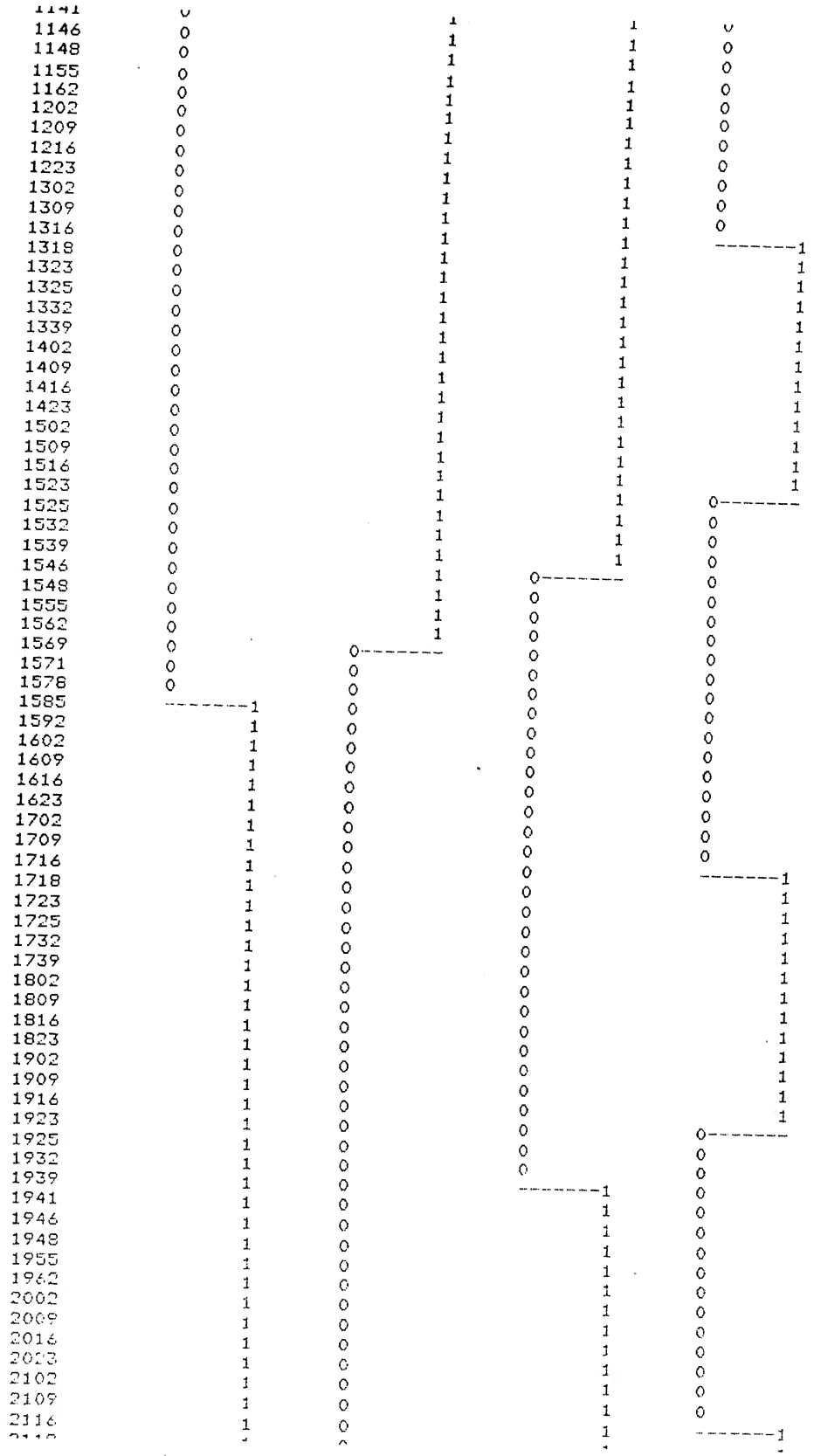
LISTING 5.2.

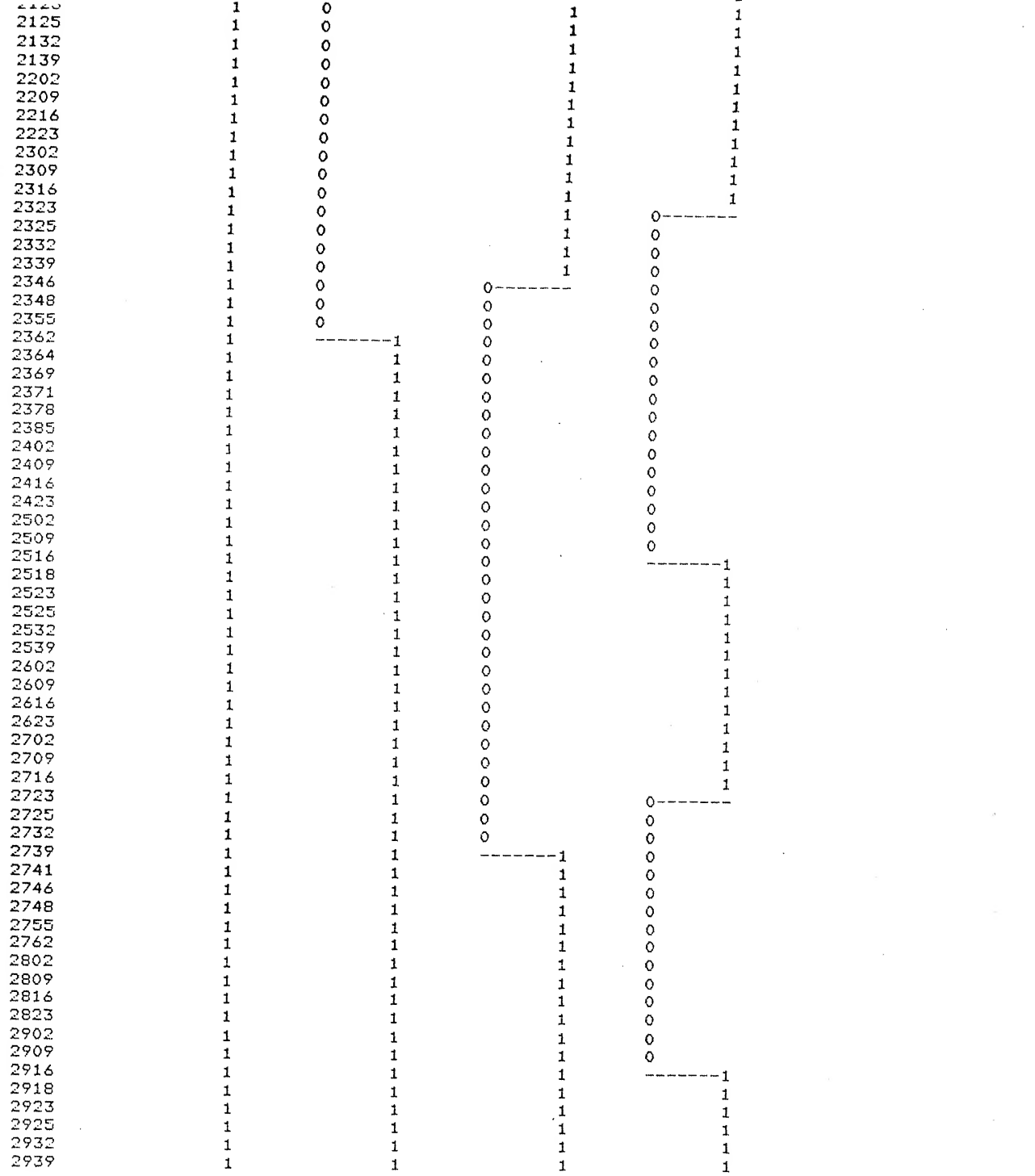
5.56
:19:59]
GIVEN TABLE IS :

INDEX	NAME	VALUE	NFO	NFI	FOL	FIL	KIND
1	INPT	1	20	0	3 4 5 6 12 12 13 13 14 15 16 17 19 19 20 20 21 22 23 24	0 0 0 0 0 0 0 0 0 0	BINP
2	CLCK	1	3	0	12 13 18 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0	BINP
3	SLVE	0	5	3	4 13 19 20 25 0 0 0 0 0	1 16 4 0 0 0 0 0 0 0	NAND
4	SLVE	1	2	3	3 12 0 0 0 0 0 0 0 0	1 17 3 0 0 0 0 0 0 0	NAND
5	SLVE	0	5	3	6 20 26 27 32 0 0 0 0 0	1 23 6 0 0 0 0 0 0 0	NAND
6	SLVE	1	2	3	5 19 0 0 0 0 0 0 0 0	1 24 5 0 0 0 0 0 0 0	NAND
7	SLVE	0	5	3	8 27 33 34 39 0 0 0 0 0	11 30 8 0 0 0 0 0 0 0	NAND
8	SLVE	1	2	3	7 26 0 0 0 0 0 0 0 0	11 31 7 0 0 0 0 0 0 0	NAND
9	SLVE	0	2	3	10 34 0 0 0 0 0 0 0 0	11 37 10 0 0 0 0 0 0 0	NAND
10	SLVE	1	2	3	9 33 0 0 0 0 0 0 0 0	11 38 9 0 0 0 0 0 0 0	NAND
11	INPT	1	20	0	7 8 9 10 26 26 27 27 28 29 30 31 33 33 34 34 35 36 37 38	0 0 0 0 0 0 0 0 0 0	BINP
12	MSTR	2	1	4	14 0 0 0 0 0 0 0 0 0	2 1 1 4 0 0 0 0 0 0	NAND
13	MSTR	2	1	4	15 0 0 0 0 0 0 0 0 0	2 1 1 3 0 0 0 0 0 0	NAND
14	MSTR	0	2	3	15 16 0 0 0 0 0 0 0 0	1 12 15 0 0 0 0 0 0 0	NAND
15	MSTR	1	2	3	14 17 0 0 0 0 0 0 0 0	1 13 14 0 0 0 0 0 0 0	NAND
16	SLVE	2	1	3	3 0 0 0 0 0 0 0 0 0	1 14 18 0 0 0 0 0 0 0	NAND
17	SLVE	2	1	3	4 0 0 0 0 0 0 0 0 0	1 15 18 0 0 0 0 0 0 0	NAND
18	NOT1	2	2	1	16 17 0 0 0 0 0 0 0 0	2 0 0 0 0 0 0 0 0 0	NOT
19	MSTR	2	1	4	21 0 0 0 0 0 0 0 0 0	3 1 1 6 0 0 0 0 0 0	NAND
20	MSTR	2	1	4	22 0 0 0 0 0 0 0 0 0	3 1 1 5 0 0 0 0 0 0	NAND
21	MSTR	0	2	3	22 23 0 0 0 0 0 0 0 0	1 19 22 0 0 0 0 0 0 0	NAND
22	MSTR	1	2	3	21 24 0 0 0 0 0 0 0 0	1 20 21 0 0 0 0 0 0 0	NAND
23	SLVE	2	1	3	5 0 0 0 0 0 0 0 0 0	1 21 25 0 0 0 0 0 0 0	NAND
24	SLVE	2	1	3	6 0 0 0 0 0 0 0 0 0	1 22 25 0 0 0 0 0 0 0	NAND
25	NOT1	2	2	1	23 24 0 0 0 0 0 0 0 0	3 0 0 0 0 0 0 0 0 0	NOT
26	MSTR	2	1	4	28 0 0 0 0 0 0 0 0 0	5 11 11 8 0 0 0 0 0 0	NAND
27	MSTR	2	1	4	29 0 0 0 0 0 0 0 0 0	5 11 11 7 0 0 0 0 0 0	NAND
28	MSTR	0	2	3	29 30 0 0 0 0 0 0 0 0	11 26 29 0 0 0 0 0 0 0	NAND
29	MSTR	1	2	3	28 31 0 0 0 0 0 0 0 0	11 27 28 0 0 0 0 0 0 0	NAND
30	SLVE	2	1	3	7 0 0 0 0 0 0 0 0 0	11 28 32 0 0 0 0 0 0 0	NAND
31	SLVE	2	1	3	8 0 0 0 0 0 0 0 0 0	11 29 32 0 0 0 0 0 0 0	NAND
32	NOT1	2	2	1	30 31 0 0 0 0 0 0 0 0	5 0 0 0 0 0 0 0 0 0	NOT
33	MSTR	2	1	4	35 0 0 0 0 0 0 0 0 0	7 11 11 10 0 0 0 0 0 0	NAND
34	MSTR	2	1	4	36 0 0 0 0 0 0 0 0 0	7 11 11 9 0 0 0 0 0 0	NAND
35	MSTR	0	2	3	36 37 0 0 0 0 0 0 0 0	11 33 36 0 0 0 0 0 0 0	NAND
36	MSTR	1	2	3	35 38 0 0 0 0 0 0 0 0	11 34 35 0 0 0 0 0 0 0	NAND
37	SLVE	2	1	3	9 0 0 0 0 0 0 0 0 0	11 35 39 0 0 0 0 0 0 0	NAND
38	SLVE	2	1	3	10 0 0 0 0 0 0 0 0 0	11 36 39 0 0 0 0 0 0 0	NAND
39	NOT1	2	2	1	37 38 0 0 0 0 0 0 0 0	7 0 0 0 0 0 0 0 0 0	NOT

evaluation results

TIME=	GATE 9	GATE 7	GATE 5	GATE 3
*****	********	********	********	********
0	0	0	0	0
102	0	0	0	0
109	0	0	0	0
116	0	0	0	1
118	0	0	0	1
123	0	0	0	1
125	0	0	0	1
132	0	0	0	1





TOTAL TIME DELAY= 3000

THE EXECUTION TIME : 12.855 SECONDS

FIGURE 5.5 AND FIGURE 5.6.

```
 TY SYNCNT
[33.24]
* THIS CIRCUIT DESCRIBES SYNCHRONOUS COUNTER.
* THERE IS ONE SUBCIRCUIT DESCRIPTION(J-K FLIP FLOP).
* THERE ARE TOTALLY 44 ELEMENTS.
* INTIALLY 16 ELEMENTS ARE GLOBALLY NUMBERED.
*FLIP FLOP DESCRIPTION IS SAME AS THAT IN RIPPLE COUNTER.
FLFP 14 7 5 1 2 3 4 5 12 13
MSTR 6 2 4 1 2 5 13 NAND 9
MSTR 7 2 4 1 3 4 12 NAND 9
MSTR 8 0 3 4 6 9 NAND 7
MSTR 9 1 3 5 7 8 NAND 7
SLVE 10 2 3 5 8 14 NAND 7
SLVE 11 2 3 4 9 14 NAND 7
SLVE 12 0 3 4 10 13 NAND 0
SLVE 13 1 3 5 11 12 NAND 0
NOT1 14 2 1 1 NOT 2
ENDS
CLCK 1 1 0 BINP 0
* INDEX NO. OF CLOCK IS 1.
CNEN 2 1 0 BINP 0
AND1 11 2 2 2 3 AND 5
AND2 12 2 2 11 4 AND 5
AND3 13 2 2 12 5 AND 5
AND4 14 2 2 13 6 AND 5
HIGH 15 1 0 BINP 0
HIGH 16 1 0 BINP 0
* 15 & 16 ARE CONNECTED TO PRESET AND CLEAR OF FLIP FLOP.
ENDM
CALL FLFP 1 2 2 15 15 3 7
CALL FLFP 1 11 11 15 15 4 8
CALL FLFP 1 12 12 16 16 5 9
CALL FLFP 1 13 13 16 16 6 10
* HERE FLIP FLOP IS CALLED 4 TIMES.
ENDC
0
```

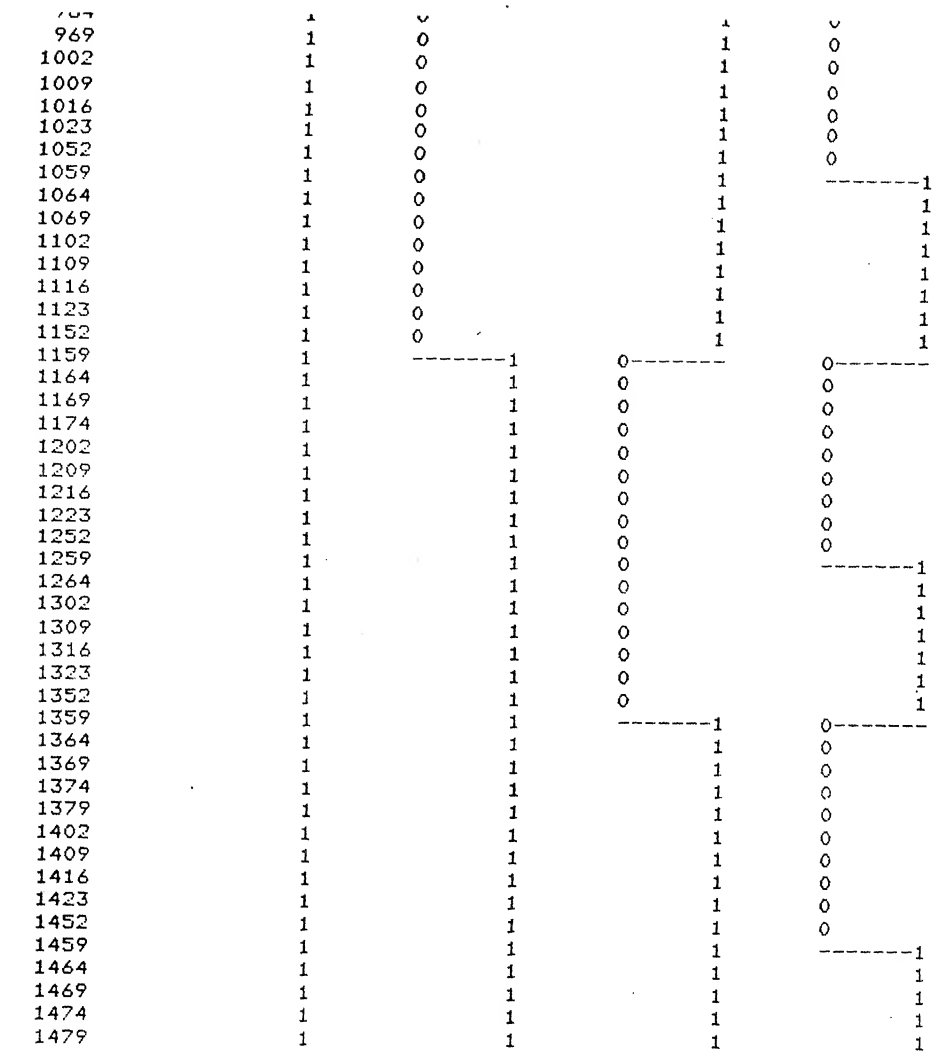
LISTING 5.3.

5.8
37:01]
GIVEN TABLE IS :

NDEX	NAME	VALUE	NFO	NFI	FOL										FIL										KIND	
1	CLCK	1	12	0	17	18	23	24	25	30	31	32	37	38	0	0	0	0	0	0	0	0	0	0	BINP	
2	CNEN	1	3	0	39	44	0	0	0	0	0	0	0	0												
3	SLVE	0	3	3	11	17	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BINP	
4	SLVE	0	3	3	7	11	18	0	0	0	0	0	0	0	15	21	7	0	0	0	0	0	0	0	NAND	
5	SLVE	0	3	3	8	12	25	0	0	0	0	0	0	0	15	28	8	0	0	0	0	0	0	0	NAND	
6	SLVE	0	3	3	9	13	32	0	0	0	0	0	0	0	16	35	9	0	0	0	0	0	0	0	NAND	
7	SLVE	1	2	3	10	14	39	0	0	0	0	0	0	0	16	42	10	0	0	0	0	0	0	0	NAND	
8	SLVE	1	2	3	3	17	0	0	0	0	0	0	0	0	15	22	3	0	0	0	0	0	0	0	NAND	
9	SLVE	1	2	3	4	24	0	0	0	0	0	0	0	0	15	29	4	0	0	0	0	0	0	0	NAND	
10	SLVE	1	2	3	5	31	0	0	0	0	0	0	0	0	16	36	5	0	0	0	0	0	0	0	NAND	
11	AND1	2	3	2	6	38	0	0	0	0	0	0	0	0	16	43	6	0	0	0	0	0	0	0	NAND	
12	AND2	2	3	2	12	24	25	0	0	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	NAND	
13	AND3	2	3	2	13	31	32	0	0	0	0	0	0	0	11	4	0	0	0	0	0	0	0	0	AND	
14	AND4	2	0	2	14	38	39	0	0	0	0	0	0	0	12	5	0	0	0	0	0	0	0	0	AND	
15	HIGH	1	16	0	0	0	0	0	0	0	0	0	0	0	13	6	0	0	0	0	0	0	0	0	AND	
16	HIGH	1	16	0	3	4	7	8	17	18	19	20	21	22	0	0	0	0	0	0	0	0	0	0	AND	
17	MSTR	2	1	4	24	25	26	27	28	29	0	0	0	0												
18	MSTR	2	1	4	5	6	9	10	31	32	33	34	35	36	0	0	0	0	0	0	0	0	0	0	BINP	
19	MSTR	0	2	3	38	39	40	41	42	43	0	0	0	0												
20	MSTR	1	2	3	19	0	0	0	0	0	0	0	0	0	1	2	15	7	0	0	0	0	0	0	NAND	
21	SLVE	2	1	3	20	0	0	0	0	0	0	0	0	0	1	2	15	3	0	0	0	0	0	0	NAND	
22	SLVE	2	1	3	20	21	0	0	0	0	0	0	0	0	15	17	20	0	0	0	0	0	0	0	NAND	
23	NOT1	2	2	1	19	22	0	0	0	0	0	0	0	0	15	18	19	0	0	0	0	0	0	0	NAND	
24	MSTR	2	1	4	3	0	0	0	0	0	0	0	0	0	15	19	23	0	0	0	0	0	0	0	NAND	
25	MSTR	2	1	4	7	0	0	0	0	0	0	0	0	0	15	20	23	0	0	0	0	0	0	0	NAND	
26	MSTR	0	2	3	21	22	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NOT	
27	MSTR	1	2	3	26	0	0	0	0	0	0	0	0	0	1	11	15	8	0	0	0	0	0	0	NAND	
28	SLVE	2	1	3	27	0	0	0	0	0	0	0	0	0	1	11	15	4	0	0	0	0	0	0	NAND	
29	SLVE	2	1	3	27	28	0	0	0	0	0	0	0	0	15	24	27	0	0	0	0	0	0	0	NAND	
30	NOT1	2	2	1	26	29	0	0	0	0	0	0	0	0	15	25	26	0	0	0	0	0	0	0	NAND	
31	MSTR	2	1	4	4	0	0	0	0	0	0	0	0	0	15	26	30	0	0	0	0	0	0	0	NAND	
32	MSTR	2	1	4	8	0	0	0	0	0	0	0	0	0	15	27	30	0	0	0	0	0	0	0	NAND	
33	MSTR	0	2	3	28	29	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NOT	
34	MSTR	1	2	3	33	0	0	0	0	0	0	0	0	0	1	12	16	9	0	0	0	0	0	0	NAND	
35	SLVE	2	1	3	34	0	0	0	0	0	0	0	0	0	1	12	16	5	0	0	0	0	0	0	NAND	
36	SLVE	2	1	3	34	35	0	0	0	0	0	0	0	0	16	31	34	0	0	0	0	0	0	0	NAND	
37	NOT1	2	2	1	33	36	0	0	0	0	0	0	0	0	16	32	33	0	0	0	0	0	0	0	NAND	
38	MSTR	2	1	4	5	0	0	0	0	0	0	0	0	0	16	33	37	0	0	0	0	0	0	0	NAND	
39	MSTR	2	1	4	9	0	0	0	0	0	0	0	0	0	16	34	37	0	0	0	0	0	0	0	NAND	
40	MSTR	0	2	3	35	36	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NOT	
41	MSTR	1	2	3	40	0	0	0	0	0	0	0	0	0	1	13	16	10	0	0	0	0	0	0	NAND	
42	SLVE	2	1	3	41	0	0	0	0	0	0	0	0	0	1	13	16	6	0	0	0	0	0	0	NAND	
43	SLVE	2	1	3	40	43	0	0	0	0	0	0	0	0	16	38	41	0	0	0	0	0	0	0	NAND	
44	NOT1	2	2	1	6	0	0	0	0	0	0	0	0	0	16	39	40	0	0	0	0	0	0	0	NAND	
					10	0	0	0	0	0	0	0	0	0	16	40	44	0	0	0	0	0	0	0	NAND	
					42	43	0	0	0	0	0	0	0	0	16	41	44	0	0	0	0	0	0	0	NAND	
															1	0	0	0	0	0	0	0	0	0	NOT	

evaluation results

ME=	GATE 6	GATE 5	GATE 4	GATE 3
***	********	********	********	********
0	0	0	0	0
2	0	0	0	0
3	0	0	0	



TOTAL TIME DELAY= 1500

THE EXECUTION TIME : 7.490 SECONDS

FIGURE 5.8.

```
 TY INPUT
[23:59:16]
* THIS IS 4 BIT ADDER/SUBTRACTOR DESCRIPTION.
* THERE ARE TOTALLY 77 ELEMENTS.
* 21 ELEMENTS ARE GLOBALLY NUMBERED.
* THE FULL ADDER CIRCUIT OF LISTING 5.1
* IS DESCRIBED AS SUBCIRCUIT HERE.
FADR 19 5 3 1 2 7 15 16
* SUBCIRCUIT HAS 19 ELEMENTS.
* 5 EXTERNAL NODES & 3 INPUT NODES.
* INPUT A IS 1,B IS 2 & CIN IS 3.
* SUM OUTPUT IS 16 & CARRY OUTPUT IS 15.
INPT 1 1 0 BINP 0
INPT 2 0 0 BINP 0
INVT 3 1 1 2 NOT 3
INVT 4 0 1 1 NOT 3
NOR1 5 2 2 1 2 NOR 6
NOR2 6 2 2 3 4 NOR 6
CINB 7 1 0 BINP 0
NOR3 8 2 2 5 6 NOR 6
BUF1 9 2 2 7 8 BFFR 6
BUF2 10 2 2 11 5 BFFR 6
LOWI 11 0 0 BINP 0
BUF3 12 2 2 13 6 BFFR 6
HIGH 13 1 0 BINP 0
* HIGH INDICATES LOGICAL 1.
* LOW INDICATES LOGICAL 0.
BUF4 14 2 2 7 6 BFFR 6
BUS1 15 2 3 9 10 12 BUS 0
BUS2 16 2 3 14 18 19 BUS 0
INVT 17 2 1 7 NOT 3
BUF5 18 2 2 7 5 BFFR 6
BUF6 19 2 2 17 8 BFFR 6
ENDS
IPA1 1 0 0 BINP 0
IPA2 2 0 0 BINP 0
IPA3 3 0 0 BINP 0
IPA4 4 1 0 BINP 0
IPB1 5 1 0 BINP 0
IPB2 6 1 0 BINP 0
IPB3 7 0 0 BINP 0
IPB4 8 0 0 BINP 0
CINP 9 0 0 BINP 0
2COM 10 2 2 5 9 XOR 6
2COM 11 2 2 6 9 XOR 6
2COM 12 2 2 7 9 XOR 6
2COM 13 2 2 8 9 XOR 6
ENDM
CALL FADR 1 10 9 14 15
CALL FADR 2 11 14 16 17
CALL FADR 3 12 16 18 19
CALL FADR 4 13 18 20 21
* FADR SUBCIRCUIT IS CALLED 4 TIMES.
ENDC
10 1 9 1
70 3 1 1 2 1 9 0
130 2 1 0 9 1
0
```

LISTING 5.4.

TY 5.10
23:46:[illegible]7]
HE GIVEN TABLE IS :

INDEX	NAME	VALUE	NFO	NFI	FOL										FIL										K
1	IPA1	0	2	0	23	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
2	IPA2	0	2	0	37	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
3	IPA3	0	2	0	51	52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
4	IPA4	1	2	0	65	66	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
5	IPB1	1	1	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
6	IPB2	1	1	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
7	IPB3	0	1	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
8	IPB4	0	1	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
9	CINP	0	8	0	10	11	12	13	27	32	33	34	0	0	0	0	0	0	0	0	0	0	0	0	BI
10	2COM	2	2	2	22	24	0	0	0	0	0	0	0	0	5	9	0	0	0	0	0	0	0	0	XO
11	2COM	2	2	2	36	38	0	0	0	0	0	0	0	0	6	9	0	0	0	0	0	0	0	0	XO
12	2COM	2	2	2	50	52	0	0	0	0	0	0	0	0	7	9	0	0	0	0	0	0	0	0	XO
13	2COM	2	2	2	64	66	0	0	0	0	0	0	0	0	8	9	0	0	0	0	0	0	0	0	XO
14	BUS1	2	4	3	41	46	47	48	0	0	0	0	0	0	27	28	30	0	0	0	0	0	0	0	BU
15	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	0	32	34	35	0	0	0	0	0	0	0	BU
16	BUS1	2	4	3	55	60	61	62	0	0	0	0	0	0	41	42	44	0	0	0	0	0	0	0	BU
17	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	0	46	48	49	0	0	0	0	0	0	0	BU
18	BUS1	2	4	3	69	74	75	76	0	0	0	0	0	0	55	56	58	0	0	0	0	0	0	0	BU
19	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	0	60	62	63	0	0	0	0	0	0	0	BU
20	BUS1	2	0	3	0	0	0	0	0	0	0	0	0	0	69	70	72	0	0	0	0	0	0	0	BU
21	BUS2	2	0	3	0	0	0	0	0	0	0	0	0	0	74	76	77	0	0	0	0	0	0	0	BU
22	INVT	1	1	1	25	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	NO
23	INVT	0	1	1	25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	NO
24	NOR1	2	3	2	26	28	34	0	0	0	0	0	0	0	1	10	0	0	0	0	0	0	0	0	NO
25	NOR2	2	3	2	26	30	32	0	0	0	0	0	0	0	22	23	0	0	0	0	0	0	0	0	NO
26	NOR3	2	2	2	27	35	0	0	0	0	0	0	0	0	24	25	0	0	0	0	0	0	0	0	NO
27	BUF1	2	1	2	14	0	0	0	0	0	0	0	0	0	9	26	0	0	0	0	0	0	0	0	BF
28	BUF2	2	1	2	14	0	0	0	0	0	0	0	0	0	29	24	0	0	0	0	0	0	0	0	BF
29	LOWI	0	1	0	28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
30	BUF3	2	1	2	14	0	0	0	0	0	0	0	0	0	31	25	0	0	0	0	0	0	0	0	BF
31	HIGH	1	1	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
32	BUF4	2	1	2	15	0	0	0	0	0	0	0	0	0	9	25	0	0	0	0	0	0	0	0	BF
33	INVT	2	1	1	35	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	NO
34	BUF5	2	1	2	15	0	0	0	0	0	0	0	0	0	9	24	0	0	0	0	0	0	0	0	BF
35	BUF6	2	1	2	15	0	0	0	0	0	0	0	0	0	33	26	0	0	0	0	0	0	0	0	BF
36	INVT	1	1	1	39	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	NO
37	INVT	0	1	1	39	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	NO
38	NOR1	2	3	2	40	42	48	0	0	0	0	0	0	0	2	11	0	0	0	0	0	0	0	0	NO
39	NOR2	2	3	2	40	44	46	0	0	0	0	0	0	0	36	37	0	0	0	0	0	0	0	0	NO
40	NOR3	2	2	2	41	49	0	0	0	0	0	0	0	0	38	39	0	0	0	0	0	0	0	0	NO
41	BUF1	2	1	2	16	0	0	0	0	0	0	0	0	0	14	40	0	0	0	0	0	0	0	0	BF
42	BUF2	2	1	2	16	0	0	0	0	0	0	0	0	0	43	38	0	0	0	0	0	0	0	0	BF
43	LOWI	0	1	0	42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
44	BUF3	2	1	2	16	0	0	0	0	0	0	0	0	0	45	39	0	0	0	0	0	0	0	0	BF
45	HIGH	1	1	0	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
46	BUF4	2	1	2	17	0	0	0	0	0	0	0	0	0	14	39	0	0	0	0	0	0	0	0	BF
47	INVT	2	1	1	49	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0	0	0	NO
48	BUF5	2	1	2	17	0	0	0	0	0	0	0	0	0	14	38	0	0	0	0	0	0	0	0	BF
49	BUF6	2	1	2	17	0	0	0	0	0	0	0	0	0	47	40	0	0	0	0	0	0	0	0	BF
50	INVT	1	1	1	53	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	NO
51	INVT	0	1	1	53	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	NO
52	NOR1	2	3	2	54	56	62	0	0	0	0	0	0	0	3	12	0	0	0	0	0	0	0	0	NO
53	NOR2	2	3	2	54	58	60	0	0	0	0	0	0	0	50	51	0	0	0	0	0	0	0	0	NO
54	NOR3	2	2	2	55	63	0	0	0	0	0	0	0	0	52	53	0	0	0	0	0	0	0	0	NO
55	BUF1	2	1	2	18	0	0	0	0	0	0	0	0	0	16	54	0	0	0	0	0	0	0	0	BF
56	BUF2	2	1	2	18	0	0	0	0	0	0	0	0	0	57	52	0	0	0	0	0	0	0	0	BF
57	LOWI	0	1	0	56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
58	BUF3	2	1	2	18	0	0	0	0	0	0	0	0	0	59	53	0	0	0	0	0	0	0	0	BF
59	HIGH	1	1	0	58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BI
60	BUF4	2	1	2	19	0	0	0	0	0	0	0	0	0	16	53	0	0	0	0	0	0	0	0	BF
61	INVT	2	1	1	63	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	NO
62	BUF5	2	1	2	19	0	0	0	0	0	0	0	0	0	16	52	0	0	0	0	0	0	0	0	BF

67	NOR2	2	3	2	68	72	74	0	0	0	0	0	0	0	4	13	0	0	0	0	0	0
68	NOR3	2	2	2	69	77	0	0	0	0	0	0	0	0	64	65	0	0	0	0	0	0
69	BUF1	2	1	2	20	0	0	0	0	0	0	0	0	0	18	68	0	0	0	0	0	0
70	BUF2	2	1	2	20	0	0	0	0	0	0	0	0	0	71	66	0	0	0	0	0	0
71	LOWI	0	1	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
72	BUF3	2	1	2	20	0	0	0	0	0	0	0	0	0	73	67	0	0	0	0	0	0
73	HIGH	1	1	0	72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
74	BUF4	2	1	2	21	0	0	0	0	0	0	0	0	0	18	67	0	0	0	0	0	0
75	INVT	2	1	1	77	0	0	0	0	0	0	0	0	0	18	0	0	0	0	0	0	0
76	BUF5	2	1	2	21	0	0	0	0	0	0	0	0	0	18	66	0	0	0	0	0	0
77	BUF6	2	1	2	21	0	0	0	0	0	0	0	0	0	75	68	0	0	0	0	0	0

evaluation results

AT TIME=		GATE 21	GATE 19	GATE 17	GATE 15
********		********	********	********	********
0		1	0	1	1
13		1	0	1	1
16		1	0	1	1
18		1	0	1	0--------
21		1	0	1	0
23		1	0	0-------	0
**	14				
**	15				
**	16				
**	17				
26		1	0	0	0
**	20				
**	21				
28		0--------	0	0	0
31		0	0	0	--------1
33		0	--------1	0	1
73		0	1	0	1
76		0	1	0	0--------
79		0	1	0	0
82		0	1	0	0
84		0	1	0	0
**	19				
88		0	0--------	--------1	--------1
**	14				
**	15				
**	16				
**	17				
89		0	0	0--------	0--------
93		0	0	0	0
94		--------1	0	0	0
97		1	0	0	0
100		1	--------1	--------1	0
133		1	1	1	0
136		1	1	1	--------1
138		1	1	1	1
139		1	1	1	1
142		1	1	1	1
144		1	1	1	1
147		1	1	1	1
**	14				
**	15				
**	20				
**	21				
149		0--------	1	1	0--------
152		0	0--------	1	--------1
154		0	0	1	1
157		--------1	0	1	1
159		1	0	1	1
162		1	--------1	1	1
164		0--------	1	1	1

** XX :INDICATES BUS CONFLICT AT GATE/ELEMENT NO. XX .

TOTAL TIME DELAY= 200

THE EXECUTION TIME : 2.592 SECONDS

```
 TY 2903
[:23:30]
* THE MICROPROCESSOR BIT SLICER IS DESCRIBED HERE.
* 9 SUBCIRCUITS ARE DESCRIBED .
* THERE ARE TOTALLY ABOUT 990 ELEMENTS .
* INITIALLY 300 ELEMENTS ARE GLOBALLY NUMBERED.
* THIS SUBCIRCUIT FORMS RAMCELLS.
CELL 37 15 7 1 2 3 4 5 6 7 12 13 14 15 32 33 34 35
BFR1 8 2 2 1 36 BFFR 1
BFR2 9 2 2 2 36 BFFR 1
BFR3 10 2 2 3 36 BFFR 1
BFR4 11 2 2 4 36 BFFR 1
NOD1 12 2 2 8 20 NODE 0
NOD2 13 2 2 9 21 NODE 0
NOD3 14 2 2 10 22 NODE 0
NOD4 15 2 2 11 23 NODE 0
BFR5 16 2 2 12 37 BFFR 1
BFR6 17 2 2 13 37 BFFR 1
BFR7 18 2 2 14 37 BFFR 1
BFR8 19 2 2 15 37 BFFR 1
BFR9 20 2 2 28 37 BFFR 1
BF10 21 2 2 29 37 BFFR 1
BF11 22 2 2 30 37 BFFR 1
BF12 23 2 2 31 37 BFFR 1
NOT1 24 2 1 16 NOT 1
NOT2 25 2 1 17 NOT 1
NOT3 26 2 1 18 NOT 1
NOT4 27 2 1 19 NOT 1
NOT5 28 2 1 24 NOT 1
NOT6 29 2 1 25 NOT 1
NOT7 30 2 1 26 NOT 1
NOT8 31 2 1 27 NOT 1
BF13 32 2 2 12 7 BFFR 1
BF14 33 2 2 13 7 BFFR 1
BF15 34 2 2 14 7 BFFR 1
BF16 35 2 2 15 7 BFFR 1
AND1 36 2 2 5 6 AND 0
HIGH 37 1 0 BINF 0
ENDS
RMDC 24 20 4 1 2 3 4 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
* THIS SUBCIRCUIT DESCRIBES RAM-DECODER.
* THE DELAY OF DECODER IS MADE 0 BY MAKING ALL DELAYS 0.
NOT1 5 2 1 1 NOT 0
NOT2 6 2 1 2 NOT 0
NOT3 7 2 1 3 NOT 0
NOT4 8 2 1 4 NOT 0
ADR0 9 2 4 5 6 7 8 AND 0
ADR1 10 2 4 1 6 7 8 AND 0
ADR2 11 2 4 5 2 7 8 AND 0
ADR3 12 2 4 1 2 7 8 AND 0
ADR4 13 2 4 5 6 3 8 AND 0
ADR5 14 2 4 1 6 3 8 AND 0
ADR6 15 2 4 5 2 3 8 AND 0
ADR7 16 2 4 1 2 3 8 AND 0
ADR8 17 2 4 5 6 7 4 AND 0
ADR9 18 2 4 1 6 7 4 AND 0
AD10 19 2 4 5 2 7 4 AND 0
AD11 20 2 4 1 2 7 4 AND 0
AD12 21 2 4 5 6 3 4 AND 0
AD13 22 2 4 1 6 3 4 AND 0
AD14 23 2 4 5 2 3 4 AND 0
AD15 24 2 4 1 2 3 4 AND 0
ENDS
BOUT 49 33 32 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 4
* THIS SUBCIRCUIT GIVES RAM OUTPUT PORT.
BFR1 33 2 2 1 17 BFFR 1
BFR2 34 2 2 2 18 BFFR 1
BFR3 35 2 2 3 19 BFFR 1
BFR4 36 2 2 4 20 BFFR 1
BFR5 37 2 2 5 21 BFFR 1
```

```
BFR6 38 2 2 6 22 BFFR 1
BFR7 39 2 2 7 23 BFFR 1
BFR8 40 2 2 8 24 BFFR 1
BFR9 41 2 2 9 25 BFFR 1
BF10 42 2 2 10 26 BFFR 1
BF11 43 2 2 11 27 BFFR 1
BF12 44 2 2 12 28 BFFR 1
BF13 45 2 2 13 29 BFFR 1
BF14 46 2 2 14 30 BFFR 1
BF15 47 2 2 15 31 BFFR 1
BF16 48 2 2 16 32 BFFR 1
OUTB 49 2 16 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 BUS 0
ENDS
MUPX 17 9 8 1 2 3 4 14 15 16 17 13
* THIS SUBCIRCUIT DESCRIBES FUNCTIONAL BLOCK OF ALU.
BUF1 5 2 2 1 15 BFFR 1
BUF2 6 2 2 5 17 BFFR 1
BUF3 7 2 2 2 15 BFFR 1
BUF4 8 2 2 7 16 BFFR 1
BUF5 9 2 2 3 14 BFFR 1
BUF6 10 2 2 9 17 BFFR 1
BUF7 11 2 2 4 14 BFFR 1
BUF8 12 2 2 11 16 BFFR 1
BUS1 13 2 4 6 8 10 12 BUS 0
ENDS
CRCH 17 9 4 1 2 3 15 9 10 11 12 13
* THIS SUBCIRCUIT DESCRIBES CARRY CHAIN CIRCUIT.
HIGH 4 1 0 BINP 0
LOWV 5 0 0 BINP 0
BUF1 6 2 2 4 15 BFFR 1
BUF2 7 2 2 5 14 BFFR 1
BUF3 8 2 2 3 10 BFFR 1
BUS1 9 2 3 6 7 8 BUS 0
NOR1 10 2 2 17 15 NOR 1
NOT1 11 2 1 10 NOT 1
NOT2 12 2 1 13 NOT 0
NOT3 13 2 1 3 NOT 1
NOR2 14 2 2 16 15 NOR 1
INVK 16 2 1 1 NOT 0
INVF 17 2 1 2 NOT 0
ENDS
SHFT 28 12 8 1 2 3 4 5 6 7 8 13 18 23 28
* THIS DESCRIBES THE BARREL SHIFTER CIRCUIT.
BUF1 9 2 2 1 5 BFFR 1
BUF2 10 2 2 2 8 BFFR 1
BUF3 11 2 2 3 7 BFFR 1
BUF4 12 2 2 4 6 BFFR 1
BUS0 13 2 4 9 10 11 12 BUS 0
BUF5 14 2 2 1 6 BFFR 1
BUF6 15 2 2 2 5 BFFR 1
BUF7 16 2 2 3 8 BFFR 1
BUF8 17 2 2 4 7 BFFR 1
BUS1 18 2 4 14 15 16 17 BUS 0
BUF9 19 2 2 1 7 BFFR 1
BF10 20 2 2 2 6 BFFR 1
BF11 21 2 2 3 5 BFFR 1
BF12 22 2 2 4 8 BFFR 1
BUS2 23 2 4 19 20 21 22 BUS 0
BF13 24 2 2 1 8 BFFR 1
BF14 25 2 2 2 7 BFFR 1
BF15 26 2 2 3 6 BFFR 1
BF16 27 2 2 4 5 BFFR 1
BUS3 28 2 4 24 25 26 27 BUS 0
ENDS
ALDC 50 17 4 1 2 3 4 11 14 17 18 23 28 32 37 38 40 41 43 45
* THIS DESCRIBES THE MAP CIRCUIT MAPPING
* I1-I4 CONTROL INPUTS TO 12 ALU INPUTS(K,P3R) AND CIN.
NOT1 5 2 1 1 NOT 0
NOT2 6 2 1 2 NOT 0
NOT3 7 2 1 3 NOT 0
NOT4 8 2 1 4 NOT 0
AND1 9 2 3 5 3 8 AND 0
```

```
AND2 10 2 4 1 2 7 8 AND 0
IPK0 11 2 2 9 10 OR 0
AND3 12 2 3 5 2 8 AND 0
AND4 13 2 3 5 2 3 AND 0
IPK1 14 2 2 12 13 OR 0
AND5 15 2 4 1 6 7 8 AND 0
AND6 16 2 3 5 6 3 AND 0
IPK2 17 2 2 15 16 OR 0
IPK3 18 2 3 5 3 4 AND 0
AND7 19 2 3 1 6 8 AND 0
AND8 20 2 3 1 3 8 AND 0
AND9 21 2 3 3 4 5 AND 0
AN10 22 2 3 7 2 4 AND 0
IPP0 23 2 4 19 20 21 50 OR 0
AN11 24 2 2 1 2 AND 0
AN12 25 2 2 2 4 AND 0
AN13 26 2 3 5 6 3 AND 0
AN14 27 2 3 1 7 4 AND 0
IPP1 28 2 4 24 22 26 27 OR 0
AN15 29 2 3 5 4 7 AND 0
AN16 30 2 3 1 2 7 AND 0
AN17 31 2 4 1 6 3 8 AND 0
IPP2 32 2 5 29 30 31 25 13 OR 0
AN18 33 2 3 5 2 7 AND 0
AN19 34 2 3 3 4 1 AND 0
AN20 35 2 3 6 4 7 AND 0
AN21 36 2 3 1 6 7 AND 0
IPP3 37 2 5 33 34 35 36 9 OR 0
IPR0 38 2 1 18 AND 0
AN22 39 2 3 1 7 8 AND 0
IPR1 40 2 3 9 39 44 OR 0
IPR2 41 2 4 1 46 47 48 OR 0
AN23 42 2 2 1 3 AND 0
IPR3 43 2 2 4 42 OR 0
AN24 44 2 3 2 7 8 AND 0
CINP 45 2 3 49 12 15 OR 0
AN25 46 2 2 3 8 AND 0
AN26 47 2 2 7 4 AND 0
AN27 48 2 2 2 8 AND 0
AN28 49 2 2 5 3 AND 0
AN29 50 2 4 5 2 7 8 AND 0
ENDS
RSIP 26 17 9 1 3 4 5 6 7 8 9 10 19 20 21 22 23 24 25 26
* THIS DESCRIBES THE MULTIPLEXER USED FOR ALU OPERANDS.
NOT1 2 2 1 1 NOT 0
AND1 11 2 2 2 3 AND 0
AND2 12 2 2 2 4 AND 0
AND3 13 2 2 2 5 AND 0
AND4 14 2 2 2 6 AND 0
AND5 15 2 2 1 7 AND 0
AND6 16 2 2 1 8 AND 0
AND7 17 2 2 1 9 AND 0
AND8 18 2 2 1 10 AND 0
IPR0 19 2 2 11 15 OR 0
IPR1 20 2 2 12 16 OR 0
IPR2 21 2 2 13 17 OR 0
IPR3 22 2 2 14 18 OR 0
NTR0 23 2 1 19 NOT 0
NTR1 24 2 1 20 NOT 0
NTR2 25 2 1 21 NOT 0
NTR3 26 2 1 22 NOT 0
ENDS
DLCH 7 3 2 1 2 6
* THIS DESCRIBES THE D-LATCH USED FOR LATCHING Q SHIFTER OUTPUTS.
NOT1 3 2 1 1 NOT 1
FLFP 4 2 2 1 2 NAND 5
FLFP 5 2 2 2 3 NAND 5
FLFP 6 2 2 4 7 NAND 5
FLFP 7 2 2 5 6 NAND 5
ENDS
IPDO 1 1 0 BINP 0
* THE MAIN DESCRIPTION STARTS HERE .
IPM1 2 2 0 MINP 0
```

```
IPD1 2 0 0 BINP 0
IPD2 3 1 0 BINP 0
IPD3 4 0 0 BINP 0
INWR 5 1 0 BINP 0
* 5 IS RAM WRITE CONTROL INPUT.
IPB0 6 1 0 BINP 0
IPB1 7 0 0 BINP 0
IPB2 8 1 0 BINP 0
IPB3 9 0 0 BINP 0
IPA0 154 1 0 BINP 0
IPA1 155 0 0 BINP 0
IPA2 156 1 0 BINP 0
IPA3 157 0 0 BINP 0
OPA0 178 2 16 90 94 98 102 106 110 114 118 122 126 130 134 138 142 146 150 BUS 0
OPA1 179 2 16 91 95 99 103 107 111 115 119 123 127 131 135 139 143 147 151 BUS 0
OPA2 180 2 16 92 96 100 104 108 112 116 120 124 128 132 136 140 144 148 152 BUS 0
OPA3 181 2 16 93 97 101 105 109 113 117 121 125 129 133 137 141 145 149 153 BUS 0
* 178 -181 GIVES RAM PORT A OUTPUTS.
CLCK 182 1 0 BINP 0
* INDEX NO. OF CLOCK IS 182.
INI0 183 0 0 BINP 0
INI1 184 1 0 BINP 0
INI2 185 1 0 BINP 0
INI3 186 0 0 BINP 0
INI4 187 0 0 BINP 0
* 183-187 GIVES I0-I4 CONTROL INPUTS.
EABR 188 1 0 BINP 0
CEBR 189 0 0 BINP 0
BUF1 190 2 2 174 189 BFFR 1
BUF2 191 2 2 175 189 BFFR 1
BUF3 192 2 2 176 189 BFFR 1
BUF4 193 2 2 177 189 BFFR 1
IDA0 207 0 0 BINP 0
IDA1 208 1 0 BINP 0
IDA2 209 1 0 BINP 0
IDA3 210 0 0 BINP 0
IDB0 211 0 0 BINP 0
IDB1 212 1 0 BINP 0
IDB2 213 1 0 BINP 0
IDB3 214 0 0 BINP 0
* 207-214 DESCRIBES DA&DB EXT.INPUTS.
RMS0 263 1 0 BINP 0
RMS1 264 0 0 BINP 0
RMS2 265 0 0 BINP 0
RMS3 266 0 0 BINP 0
QRS0 267 1 0 BINP 0
QRS1 268 0 0 BINP 0
QRS2 269 0 0 BINP 0
QRS3 270 0 0 BINP 0
* 263-270 DESCRIBE RAM&Q SHIFTER CONTOL INPUTS.
RND1 279 2 2 190 211 NODE 0
RND2 280 2 2 191 212 NODE 0
RND3 281 2 2 192 213 NODE 0
RND4 282 2 2 193 214 NODE 0
BUF5 283 2 2 297 287 BFFR 0
BUF6 284 2 2 298 287 BFFR 0
BUF7 285 2 2 299 287 BFFR 0
BUF8 286 2 2 300 287 BFFR 0
OEBR 287 0 0 BINP 0
RND5 288 2 2 283 1 NODE 0
RND6 289 2 2 284 2 NODE 0
RND7 290 2 2 285 3 NODE 0
RND8 291 2 2 286 4 NODE 0
NOR1 292 2 4 288 289 290 291 NOR 1
ENDM
* SUBCIRCUIT CALLS WILL FOLLOW.
CALL RMDC 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
* B ADDRESS DECODED.
CALL CELL 288 289 290 291 5 10 158 26 27 28 29 90 91 92 93
CALL CELL 288 289 290 291 5 11 159 30 31 32 33 94 95 96 97
CALL CELL 288 289 290 291 5 12 160 34 35 36 37 98 99 100 101
CALL CELL 288 289 290 291 5 13 161 38 39 40 41 102 103 104 105
```

```
CALL CELL 288 289 290 291 5 14 162 42 43 44 45 106 107 108 109
CALL CELL 288 289 290 291 5 15 163 46 47 48 49 110 111 112 113
CALL CELL 288 289 290 291 5 16 164 50 51 52 53 114 115 116 117
CALL CELL 288 289 290 291 5 17 165 54 55 56 57 118 119 120 121
CALL CELL 288 289 290 291 5 18 166 58 59 60 61 122 123 124 125
CALL CELL 288 289 290 291 5 19 167 62 63 64 65 126 127 128 129
CALL CELL 288 289 290 291 5 20 168 66 67 68 69 130 131 132 133
CALL CELL 288 289 290 291 5 21 169 70 71 72 73 134 135 136 137
CALL CELL 288 289 290 291 5 22 170 74 75 76 77 138 139 140 141
CALL CELL 288 289 290 291 5 23 171 78 79 80 81 142 143 144 145
CALL CELL 288 289 290 291 5 24 172 82 83 84 85 146 147 148 149
CALL CELL 288 289 290 291 5 25 173 86 87 88 89 150 151 152 153
* 16*4 RAM CELLS ARE FORMED.
CALL RMDC 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173
*A ADDRESS DECODED.
CALL BOUT 26 30 34 38 42 46 50 54 58 62 66 70 74 78 82 86 10 11 12 13 14 15 16 17 18 19 20 21 22 23
CALL BOUT 27 31 35 39 43 47 51 55 59 63 67 71 75 79 83 87 10 11 12 13 14 15 16 17 18 19 20 21 22 23
CALL BOUT 28 32 36 40 44 48 52 56 60 64 68 72 76 80 84 88 10 11 12 13 14 15 16 17 18 19 20 21 22 23
CALL BOUT 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 10 11 12 13 14 15 16 17 18 19 20 21 22 23
* RAM OUTPUT PORT B (4 BITS ) ARE FORMED.
CALL ALDC 184 185 186 187 194 195 196 197 198 199 200 201 202 203 204 205 206
* MAPS I1-14 INPUTS TO 13 ALU CONTROL BITS.
CALL RSIP 188 178 179 180 181 207 208 209 210 215 216 217 218 219 220 221 222
CALL RSIP 183 279 280 281 282 293 294 295 296 223 224 225 226 227 228 229 230
* ALU OPERANDS ARE MULTIPLEXED.
CALL MUPX 198 199 200 201 215 219 223 227 231
CALL MUPX 194 195 196 197 215 219 223 227 232
CALL CRCH 232 231 206 301 233 234 235 236 237
CALL MUPX 202 203 204 205 234 235 236 237 238
* 3 MUPX & 1 CRCH GIVE 1 BIT ALU OUTPUT.
* 238 IS LSB OF ALU OUTPUT.
CALL MUPX 198 199 200 201 216 220 224 228 239
CALL MUPX 194 195 196 197 216 220 224 228 240
CALL CRCH 240 239 233 301 241 242 243 244 245
CALL MUPX 202 203 204 205 242 243 244 245 246
* 246 IS SECOND ALU OUTPUT BIT.
CALL MUPX 198 199 200 201 217 221 225 229 247
CALL MUPX 194 195 196 197 217 221 225 229 248
CALL CRCH 248 247 241 301 249 250 251 252 253
CALL MUPX 202 203 204 205 250 251 252 253 254
* 254 IS THIRD ALU OUTPUT BIT.
CALL MUPX 198 199 200 201 218 222 226 230 255
CALL MUPX 194 195 196 197 218 222 226 230 256
CALL CRCH 256 255 249 301 257 258 259 260 261
CALL MUPX 202 203 204 205 258 259 260 261 262
* 262 IS MSB OF ALU OUTPUT.
CALL SHFT 238 246 254 262 263 264 265 266 271 272 273 274
* ALU -SHIFTER OUTPUT(WHICH IS CONNECTED TO RAM INPUTS).
CALL SHFT 238 246 254 262 267 268 269 270 275 276 277 278
* Q-SHIFTER OUTPUT.
CALL DLCH 275 302 293
CALL DLCH 276 302 294
CALL DLCH 277 302 295
CALL DLCH 278 302 296
CALL DLCH 271 302 297
CALL DLCH 272 302 298
CALL DLCH 273 302 299
CALL DLCH 274 302 300
* SHIFTER OUTPUTS ARE LATCHED.
ENDC
1 2 3 0 8 0
5 2 1 0 6 0
10 2 2 1 7 1
15 2 1 1 6 1
20 2 3 1 8 1
25 2 1 0 6 0
30 2 2 0 7 0
35 2 4 1 9 1
40 2 3 0 8 0
55 2 1 1 6 1
60 2 2 1 7 1
65 2 1 0 6 0
```

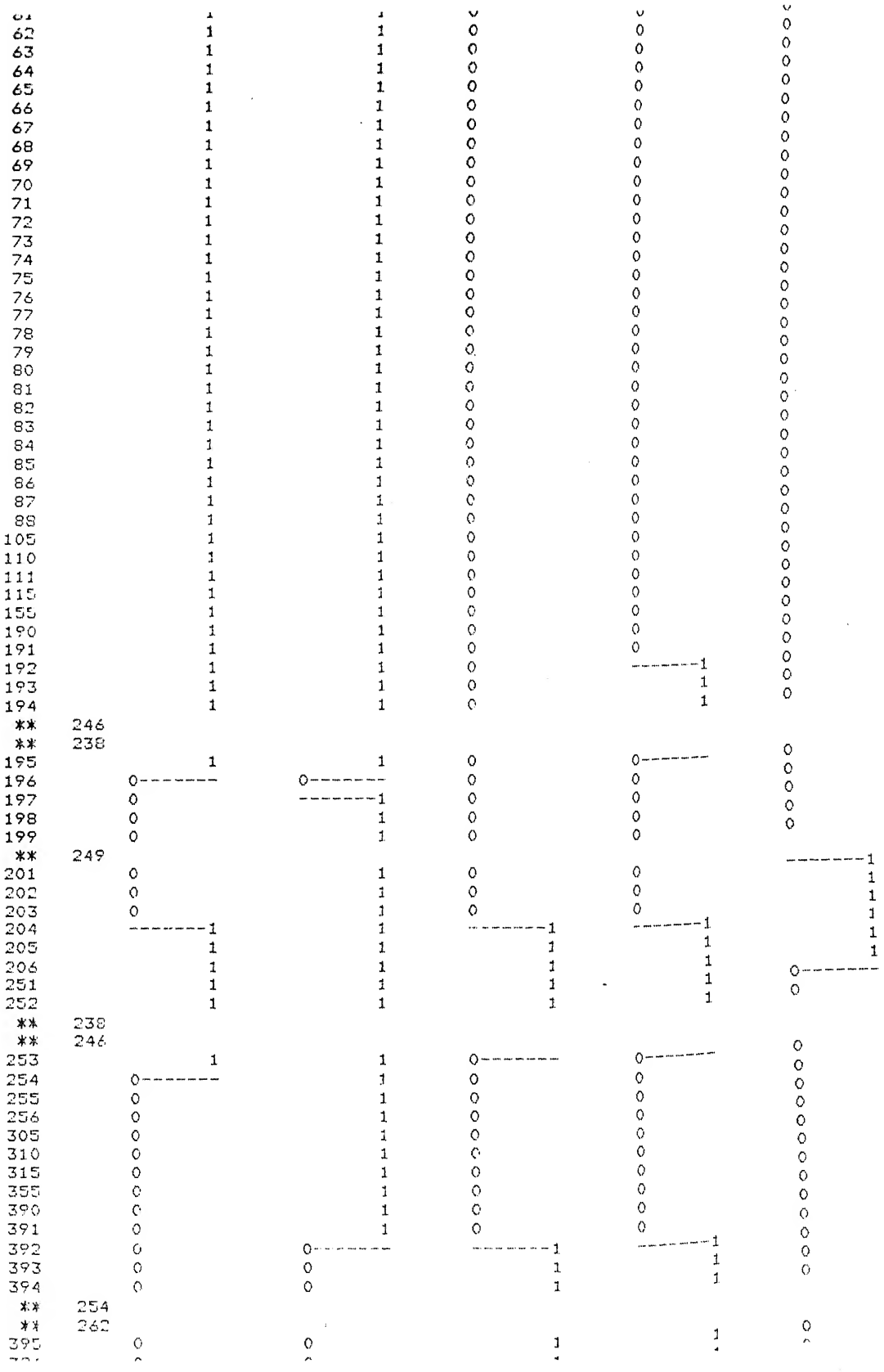
```
70 2 3 1 8 1
75 2 1 1 6 1
80 2 2 0 7 0
85 1 5 0
87 4 9 0 157 1 287 1 156 0
190 3 189 1 188 0 184 0
390 8 184 1 187 1 188 1 209 0 210 1 9 1 8 0 7 1
590 7 188 0 186 1 154 0 156 1 9 0 8 1 7 0
790 3 185 0 154 1 8 0
990 1 187 0
1190 1 185 1
1385 2 184 0 157 0
1585 2 185 0 8 1
1785 3 186 0 187 1 188 1
1985 3 184 1 188 0 183 1
2185 4 183 0 189 0 156 0 187 0
2385 5 189 1 184 0 186 1 187 1 157 1
2585 1 185 1
2785 3 185 0 186 0 187 0
0
292 11 50 8 1 1
8 262 254 246 238 197 196 195 194
```

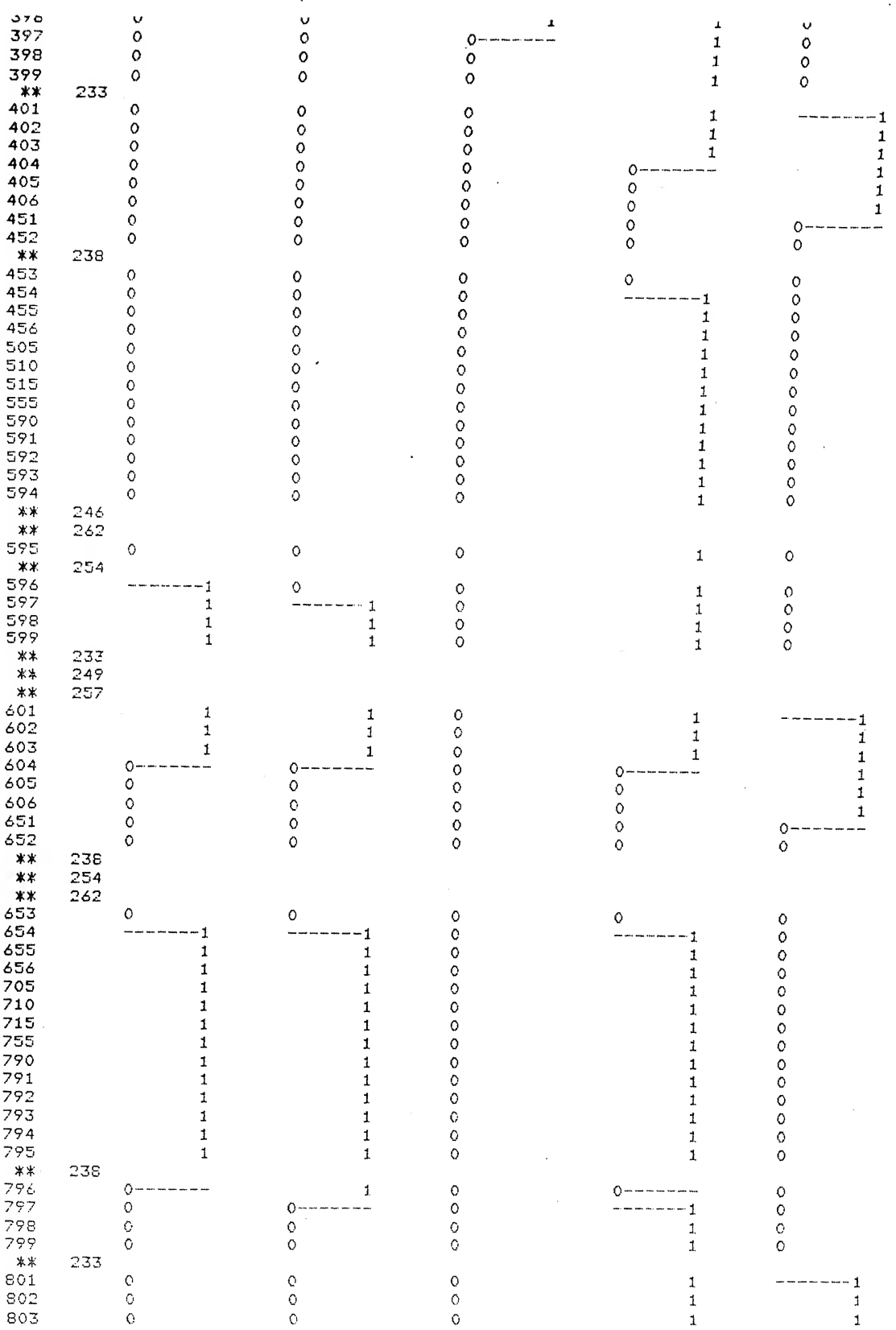
LISTING 6.1.

TY OUTPUT
[:55:47]

evaluation results

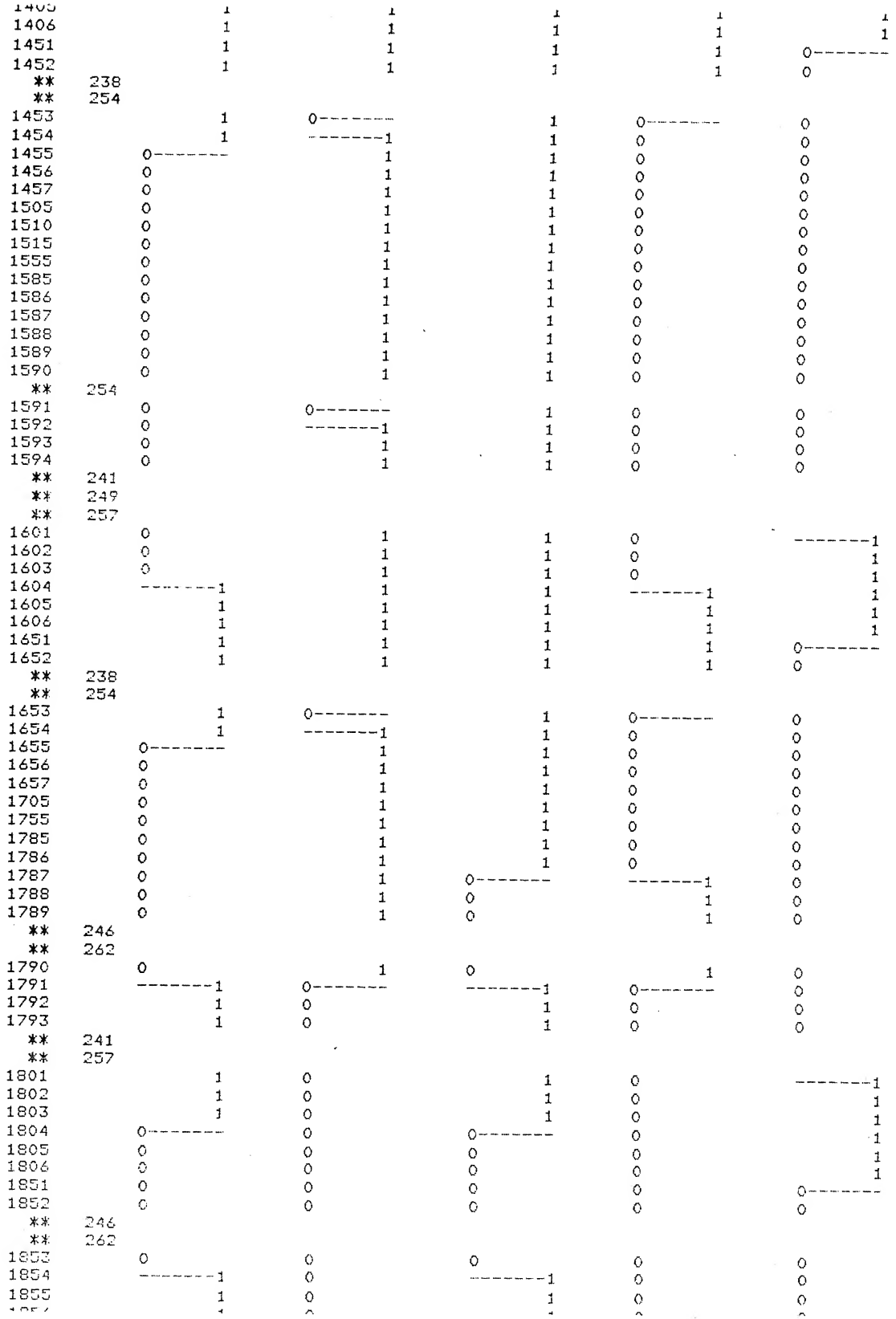
AT TIME=	GATE 262	GATE 254	GATE 246	GATE 238	GATE 301
********	********	********	********	********	********
0	1	1	1	0	1
1	1	1	1	0	1
2	1	1	1	0	1
3	1	1	1	0	1
4	1	1	1	0	1
5	1	1	1	0	1
6	1	1	1	0	1
7	1	1	1	0	1
8	1	1	1	0	1
9	1	1	1	0	1
10	1	1	1	0	1
11	1	1	1	0	1
12	1	1	1	0	1
13	1	1	1	0	1
14	1	1	1	0	1
15	1	1	1	0	1
16	1	1	1	0	1
17	1	1	1	0	1
18	1	1	1	0	1
19	1	1	1	0	1
20	1	1	1	0	1
21	1	1	1	0	1
22	1	1	1	0	1
23	1	1	1	0	1
24	1	1	1	0	1
25	1	1	1	0	1
26	1	1	1	0	1
27	1	1	1	0	1
28	1	1	1	0	1
29	1	1	1	0	1
30	1	1	1	0	1
31	1	1	1	0	1
32	1	1	1	0	1
33	1	1	1	0	1
34	1	1	1	0	1
35	1	1	1	0	1
36	1	1	1	0	1
37	1	1	1	0	1
38	1	1	1	0	1
39	1	1	1	0	1
40	1	1	1	0	1
41	1	1	1	0	1
42	1	1	1	0	1
43	1	1	1	0	1
44	1	1	1	0	1
45	1	1	1	0	1
51	1	1	1	0	0-------
52	1	1	1	0	0
53	1	1	1	0	0
54	1	1	0--------	0	0
55	1	1	0	0	0
56	1	1	0	0	0
57	1	1	0	0	0
58	1	1	0	0	0
59	1	1	0	0	0
60	1	1	0	0	0





```
804          0                 0                 0                 0--------                 1
805          0                 0                 0                 0                         1
806          0                 0                 0                 0                         1
851          0                 0                 0                 0                 0--------
852          0                 0                 0                 0                 0
 **   238
853          0                 0                 0                 0                 0
854          0                 0                 0                 --------1         0
855          0                 0                 0                           1       0
856          0                 0                 0                           1       0
905          0                 0                 0                           1       0
910          0                 0                 0                           1       0
915          0                 0                 0                           1       0
955          0                 0                 0                           1       0
990          0                 0                 0                           1       0
991          0                 0                 0                           1       0
992          0                 0                 0                           1       0
993          0                 0                 0                           1       0
994          0                 0                 0                           1       0
 **   246
 **   254
 **   262
995          0                 0                 0                           1       0
996          --------1         --------1         --------1         0--------       0
997                    1                 1                 1         0               0
998                    1                 1                 1         0               0
 **   241
 **   249
 **   257
1001                   1                 1                 1         0               --------1
1002                   1                 1                 1         0                       1
1003                   1                 1                 1         0                       1
1004         0--------         0--------         0--------         0                       1
1005         0                 0                 0                 0                       1
1006         0                 0                 0                 0                       1
1051         0                 0                 0                 0                 0--------
1052

1203         0                 0                           1       0                       1
1204         0                 0                 0--------         0                       1
1205         0                 0                 0                 0                       1
1206         0                 0                 0                 0                       1
1251         0                 0                 0                 0                 0--------
1252         0                 0                 0                 0                 0
 **   246
1253         0                 0                 0                 0                 0
1254         0                 0                 --------1         0                 0
1255         0                 0                           1       0                 0
1256         0                 0                           1       0                 0
1305         0                 0                           1       0                 0
1310         0                 0                           1       0                 0
1315         0                 0                           1       0                 0
1355         0                 0                           1       0                 0
1385         0                 0                           1       0                 0
1386         0                 0                           1       0                 0
1387         --------1         --------1         0--------         --------1         0
1388                   1                 1         0                         1       0
1389                   1                 1         0                         1       0
 **   238
 **   254
 **   262
1390         0--------         0--------         0                 0--------         0
1391         0                 --------1         --------1         0                 0
1392         0                           1                 1       0                 0
1393         0                           1                 1       0                 0
 **   241
 **   249
 **   257
1401         0                           1                 1       0                 --------1
1402         0                           1                 1       0                         1
1403         0                           1                 1       0                         1
1404         --------1                   1                 1       ----------1               1
```


1405 1 1 1 1 1
1406 1 1 1 1 1
1451 1 1 1 1 0-------
1452 1 1 1 1 0
** 238
** 254
1453 1 0------- 1 0------- 0
1454 1 -------1 1 0 0
1455 0------- 1 1 0 0
1456 0 1 1 0 0
1457 0 1 1 0 0
1505 0 1 1 0 0
1510 0 1 1 0 0
1515 0 1 1 0 0
1555 0 1 1 0 0
1585 0 1 1 0 0
1586 0 1 1 0 0
1587 0 1 1 0 0
1588 0 1 1 0 0
1589 0 1 1 0 0
1590 0 1 1 0 0
** 254
1591 0 0------- 1 0 0
1592 0 -------1 1 0 0
1593 0 1 1 0 0
1594 0 1 1 0 0
** 241
** 249
** 257
1601 0 1 1 0 -------1
1602 0 1 1 0 1
1603 0 1 1 0 1
1604 --------1 1 1 -------1 1
1605 1 1 1 1 1
1606 1 1 1 1 1
1651 1 1 1 1 0-------
1652 1 1 1 1 0
** 238
** 254
1653 1 0------- 1 0------- 0
1654 1 -------1 1 0 0
1655 0------- 1 1 0 0
1656 0 1 1 0 0
1657 0 1 1 0 0
1705 0 1 1 0 0
1755 0 1 1 0 0
1785 0 1 1 0 0
1786 0 1 1 0 0
1787 0 1 0------- -------1 0
1788 0 1 0 1 0
1789 0 1 0 1 0
** 246
** 262
1790 0 1 0 1 0
1791 --------1 0------- -------1 0------- 0
1792 1 0 1 0 0
1793 1 0 1 0 0
** 241
** 257
1801 1 0 1 0 -------1
1802 1 0 1 0 1
1803 1 0 1 0 1
1804 0-------- 0 0------- 0 1
1805 0 0 0 0 1
1806 0 0 0 0 1
1851 0 0 0 0 0-------
1852 0 0 0 0 0
** 246
** 262
1853 0 0 0 0 0
1854 --------1 0 --------1 0 0
1855 1 0 1 0 0

```
1858                1       0                1       0       0
1905                1       0                1       0       0
1910                1       0                1       0       0
1915                1       0                1       0       0
1955                1       0                1       0       0
1985                1       0                1       0       0
1986                1       0                1       0       0
1987                1       0                1       0       0
2001                1       0                1       0       -------1
2002                1       0                1       0              1
2003                1       0                1       0              1
2004        0-------        0        0-------        0              1
2005        0               0        0               0              1
2006        0               0        0               0              1
2051        0               0        0               0       0-------
2052        0               0        0               0       0
  **   246
  **   262
2053        0               0        0               0       0
2054        -------1        0        -------1        0       0
2055               1        0                1       0       0
2056               1        0                1       0       0
2105               1        0                1       0       0
2155               1        0                1       0       0
2185               1        0                1       0       0
2186               1        0                1       0       0
2187        0-------        -------1 0-------        -------1 0
2188        0                      1 0                      1 0
  **   238
  **   254
2189        0               0-------        0       0-------        0
  **   262
2190        0               0        -------1        0       0
2191        0               0        0-------        -------1 0
2192        0               -------1 0                      1 0
2193        0                      1 0                      1 0
2194        0                      1 0                      1 0
  **   233
2201        0                      1 0                      1 -------1
2202        0                      1 0                      1              1
2203        0                      1 0                      1              1
2204        -------1               1 -------1               1              1
2205               1               1        1               1              1
2206               1               1        1               1              1
2251               1               1        1               1       0-------
2252               1               1        1               1       0
  **   262
2253        0-------               1                1       1       0
2254        0                      1 0-------               1       0
2255        0                      1 0                      1       0
2256        0                      1 0                      1       0
2305        0                      1 0                      1       0
2310        0                      1 0                      1       0
2315        0                      1 0                      1       0
2355        0                      1 0                      1       0
2385        0                      1 0                      1       0
2386        0                      1 0                      1       0
2387        -------1        0------- -------1        0------- 0
2388               1        0                1       0       0
2389               1        0                1       0       0
  **   246
  **   254
2390               1        0        0-------        0       0
2391               1        0        0               0       0
2392               1        0        0               0       0
  **   233
  **   241
  **   249
  **   257
2401               1        0        0               0       -------1
2402               1        0        0               0              1
2403               1        0        0               0              1
2404        0               0        0               0              1
```

```
2404          0--------       0               0               0                       1
2405          0               0               0               0                       1
2406          0               0               0               0                       1
2451          0               0               0               0               0--------
2452          0               0               0               0               0
  **   246
  **   254
2453          0               0               0               0               0
2454          0               --------1       0               0               0
2455          0               0--------       0               0               0
2456          --------1       0               0               0               0
2457                  1       0               0               0               0
2458                  1       0               0               0               0
2505                  1       0               0               0               0
2510                  1       0               0               0               0
2515                  1       0               0               0               0
2555                  1       0               0               0               0
2585                  1       0               0               0               0
2586                  1       0               0               0               0
2587                  1       0               0               0               0
2588                  1       0               0               0               0
2589                  1       0               0               0               0
  **   262
2590          0--------       0               0               0               0
2591          0               --------1       0               0               0
2592          0                       1       0               0               0
2593          0                       1       0               0               0
  **   233
  **   241
  **   249
  **   257
2601          0                       1       0               0       --------1
2602          0                       1       0               0               1
2603          0                       1       0               0               1
2604          0               0--------       0               0               1
2605          0               0               0               0               1
2606          0               0               0               0               1
2651          0               0               0               0       0--------
2652          0               0               0               0       0
  **   246
  **   262
2653          0               0               0               0       0
2654          0               0               0               0       0
2655          0               --------1       0               0       0
2656          0                       1       0               0       0
2657          0                       1       0               0       0
2705          0                       1       0               0       0
2710          0                       1       0               0       0
2715          0                       1       0               0       0
2755          0                       1       0               0       0
2785          0                       1       0               0       0
2786          0                       1       0               0       0
2787          0               0--------       0               0       0
2788          0               0               0               0       0
2789          0               0               0               0       0
2790          0               0               0               0       0
2791          0               0               0               0       0
2801          0               0               0               0       --------1
2802          0               0               0               0               1
2803          0               0               0               0               1
2851          0               0               0               0       0--------
```

** XX :INDICATES BUS CONFLICT AT GATE/ELEMENT NO. XX .

TOTAL TIME DELAY= 2900 THE EXECUTION TIME : 71.887 SECONI

98562

Date Slip

This book is to be returned on the date last stamped.

.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...
.....	.....	...